Reverse Adaptive Krill Herd Locally Weighted Support Vector Regression for forecasting and trading Exchange Traded Funds

by

Georgios Sermpinis*

Charalampos Stasinakis**

Arman Hassanniakalager***

March 2017

Abstract

This study introduces a Reverse Adaptive Krill Herd - Locally Weighted Support Vector Regression (RKH-LSVR) model. The Reverse Adaptive Krill Herd (RKH) algorithm is a novel metaheuristic optimization technique inspired by the behaviour of krill herds. In RKH-LSVR, the RKH optimizes the locally weighted Support Vector Regression (LSVR) parameters by balancing the search between local and global optima. The proposed model is applied to the task of forecasting and trading six ETF stocks on a daily basis over the period 2010-2015. The RKH-LSVR's efficiency is benchmarked against a set of traditional SVR structures and simple linear and non-linear models. The trading application is designed in order to validate the robustness of the algorithm under study and to provide empirical evidence in favour of or against the Adaptive Market Hypothesis (AMH). In terms of the results, the RKH-LSVR outperforms its counterparts in terms of statistical accuracy and trading efficiency, while the time varying trading performance of the models under study validates the AMH theory.

Keywords

Forecasting, Support Vector Regression, Krill Herd, Adaptive Market Hypothesis, Optimization

* University of Glasgow Business School, University of Glasgow, Gilbert Scott Building, Glasgow G12 8QQ, United Kingdom (Georgios.Sermpinis@glasgow.ac.uk)
**University of Glasgow Business School, University of Glasgow, Gilbert Scott Building, Glasgow G12 8QQ, United Kingdom (Charalampos.Stasinakis@glasgow.ac.uk)
*** Research Associate, University of Glasgow Business School, University of Glasgow, Gilbert Scott Building, Glasgow G12 8QQ, United Kingdom (a.hassanniakalager.1@research.gla.ac.uk)

1. Introduction

In the competitive financial world, identifying optimal solutions for a given financial problem is very important, but it can also prove utterly challenging. The financial task might not be well defined or might suffer from lack of necessary data. Even if this is not the case, simple constraints can often impede closed form solutions or the application of standard numerical methods (Lo, 2000; Gilli *et al.*, 2008). This usually can be overcome by restating the problem through relaxed assumptions that simplify it and consequently its solution. Logically, it should be preferable to adapt an optimization approach around the original problem. This can be achieved by heuristic optimization techniques (Hall and Posner, 2007). Their success in financial applications is well documented (Chang *et al.*, 2000; Gilli *et al.*, 2008; Oreski and Oreski, 2014; Aguilar-Rivera *et al.*, 2015).

Nonetheless, their use can be computationally demanding compared to traditional models while they can trap in local optima and suffer from over-fitting. Metaheuristics are problem-dependent techniques that can overcome these issues to some extent (Turmon 1998; Bernstein *et al.*, 2015). They achieve a trade-off between intensification (local search) and randomization (global search) by intelligent selection of random variables, without being problem dependent (Talbi, 2009). Martí *et al.* (2013) compare several heuristics and metaheuristic models and suggest that the best solutions are provided with metaheuristic approaches, while computational time is also decreased. This feature made them a popular approach for solving complex optimization problems. Their modeling is inspired by activities appearing in nature. For example, many algorithms evolve around species evolution and movement (Goldberg, 1989; Yang and Gandomi, 2012; Li *et al.*, 2014) while others are based on swarm behavior and intelligence (Liang *et al.*, 2006; Karaboga and Basturk, 2008; Yang, 2010; Popescu and Crama (2015)).

A novel bio-inspired metaheuristic method is the Krill Herd algorithm (KH) as proposed by Gandomi and Alavi (2012). KH simulates the herding behavior of krill individuals. Its objective functions are the minimum distances of krill from the food location and the location of the highest density of the herd. Each krill position is a time-dependent function formulated by three motions. The movement induced by other individuals, the foraging motion and a random physical diffusion. In KH the derivative information is not necessary because it uses a stochastic random search rather than a gradient search. Additionally, KH requires the fine tuning of only one parameter, in contrary to other metaheuristics algorithms (such as the particle swarm optimization and the harmony search). Gandomi and Alavi (2012) and Wang *et al.* (2014) compare the efficiency of KH against the most popular metaheuristics optimization models. In both studies, the KH present a superior performance. It is worth noting that there is no application of the KH in a financial forecasting framework. In this paper, we extend the KH algorithm by combining the principles of opposition based optimization along with a two-stage adaptive operator (CPO) and introduce the Reverse-adaptive KH (RKH) algorithm. RKH provides a substantial improvement to the search ability of the KH for best solutions while at the same time adds diversity to the generated population. As a consequence, krill move smoother and quicker towards better solutions.

Additionally, the CPO decreases substantially the probability of trapping to local optima and enhances the efficiency between krill exploring and exploiting the search space.

Support Vector Regressions (SVRs) are non-linear data-adaptive regression techniques. SVRs main advantage is their ability to generate nonlinear decision boundaries through linear classifiers, while having a simple geometric interpretation (Suykens, 2002). SVR applications exhibit promising results in financial forecasting tasks (see amongst others Trafalis and Ince (2000), Hsu *et al.* (2009), Yeh *et al.* (2011) and Sermpinis *et al.* (2015)). Their main drawback, though, is the sensitivity to their parametrization process. For that reason, evolutionary techniques, especially GAs, are commonly combined with SVR in order to form superior forecasting hybrid structures. For example, Pai *et al.* (2006) apply epsilon SVR with genetically optimized parameters (GA-εSVR) in forecasting exchange rates. Huang *et al.* (2010) forecast the EUR/USD, GBP/USD, NZD/USD, AUD/USD, JPY/USD and RUB/USD exchange rates with a hybrid chaos-based SVR model. In their application, they confirm the forecasting superiority of their proposed model compared to chaos-based NNs and several traditional non-linear models. Lin and Pai (2010) introduce a fuzzy SVR model for forecasting indices of business cycles. Yuan (2012) suggests that (GA-εSVR) is more efficient than traditional SVR and NN models, when applied to the task of forecasting sales volume.

A more complicated but also promising class of SVRs is the Locally weighted SVR (LSVR). The conceptual advantage of the LSVR against the traditional SVR is the better balance between training error and model complexity. This is achieved by penalizing past data, while exploiting more the information from recent observations. This adaptive feature seems advantageous for financial time series. For example, Huang *et al.* (2006) apply LSVR to the task of forecasting three stock indices. Yang *et al.* (2009) apply a localized SVR forecast financial data and their results are superior to the standard SVR. Wu and Akbarov (2011) apply successfully weighted SVRs to the task of forecasting warranty claims. Finally, Jiang and He (2012) propose a hybrid SVR that incorporates the Grey relational grade weighting function. When applied to financial time series forecasting, the local Grey SVR outperforms locally weighted counterparts in terms of computational speed and prediction accuracy.

The above background motivates us to introduce the hybrid Reverse Adaptive Krill Herd - Locally Weighted Support Vector Regression (RKH-LSVR) algorithm. The RKH algorithm tunes the three parameters of the LSVR. This hybrid structure, that combines the advantages of RKH and LSVR, does not exist in the literature. In the few studies that employ LSVR the tuning of the LSVR is utilized through cross validation or grid search. Previous studies in financial forecasting problems either use traditional optimized SVRs (Trafalis and Ince, 2000; Hsu *et al.*, 2009) or apply a simple GA in the SVR's parametrization (Pai *et al.*, 2006; Wu *et al.*, 2007; Yuan, 2012). In this study, the most popular SVR techniques applied in the relevant literature will act as benchmark to the proposed RKH-LSVR algorithm. Namely, the performance of the RKH-LSVR is compared against a LSVR optimized through

KH, a SVR optimized through a GA, a SVR optimized through grid search, a set of non-linear and linear models and a random walk (RW).

All models are applied to the task of forecasting and trading seven Exchange-Traded Funds (ETFs) on a daily basis over the period 2010-2015. ETFs offer investors the opportunity to trade stock indices with low transaction costs. The six ETFs under study track some of the most liquid stock indices, currency baskets and commodities. In the GA and KH models the practitioner can choose the metaheuristics fitness function. In the literature, practitioners usually choose the Mean Square Error (MSE) or the Root Mean Square Error (RMSE). However, in financial trading applications statistical accuracy is not always synonymous of trading profitability. In this study, three different fitness functions will be explored incorporating statistical and trading terms.

The forecasting and trading exercise covers the European sovereign debt crisis. Its aim is to test the performance of the models under study when the markets are in crises, to explore their robustness within the out-of-sample and to provide empirical evidence in favour of Adaptive Market Hypothesis (AMH), as presented by Lo (2004). For this reason, four forecasting and trading exercises over the period of 2010-2015 are conducted. In each forecasting exercise, the models under study are evaluated in a monthly basis. AMH suggests that the trading models' strength varies depending on market conditions and that the performance of trading rules dies out through time. It also suggests that in periods of financial crises it is more difficult to generate profitable trading rules and that this task is even more difficult when the underlying market is advanced. The monthly evaluation of our models, the out of sample periods and the markets under study will allow us to check these elements of the AMH.

In terms of the empirical results, the RKH-LSVR outperforms its benchmarks in terms forecasting accuracy and trading profitability. The implementation of KHs and GAs in the SVR structures is beneficial in comparison with those applying data driven parametrization. The majority of the SVR models produce profitable forecasts after transaction costs, but their success seems sensitive to the parameter optimization and the periods under study. We note that the average trading performance of all models appears worse in the second forecasting exercise, while the profitability of all models deteriorates generally through time. The models generate lower profits for the currency ETFs and higher for the commodity ETFs.

The rest of the paper is organized as follows. Section 2 provides a detailed description of the dataset while a theoretical background on SVR and LSVR is given in section 3. The RKH-LSVR algorithm and its benchmarks follow in section 4. The statistical and trading performance of all models is presented in sections 5 and 6 respectively while the concluding remarks are provided in section 8. Finally, the technical characteristics of the models under study are included in the appendix section.

2. Dataset

In this analysis we examine six ETFs that are designed to replicate currencies and stock indices from US and Europe, along with two major commodities over the periods of 2010-2012 and 2013-2015. ETFs offer investors the opportunity to trade stock market indices at very low transaction costs[1]. The advantages of ETFs over "conventional trading" are well documented by researchers (Dolvin, 2010; Marshall *et al.*, 2013), practitioners (Ferri, 2009; Wagner, 2011), analysts (e.g. ETFdb.com) and institutions (e.g. ICI, US Securities and Exchange Commission (SEC)). The details of the seven ETFs under study are presented in table 1 below.

Table 1: The ETFs under study

MARKETS	ETF	TRACKING	TICKER
US	PowerShares DB US Dollar Index Bullish Fund	Long US Dollar Futures Index (USDX)	UUP
	SPDR Dow Jones Industrial Average Trust	Dow Jones Industrial Average	DIA
EU	Guggenheim CurrencyShares British Pound Sterling Trust	GBP/USD exchange rate	FXB
	Vanguard FTSE Europe	FTSE Developed Europe Index	VGK
Commodities	United States Oil Fund	Light Sweet Crude Oil (WTI) Futures Index	USO
	iShares Gold Trust	Gold Bullion	IAU

The six ETFs time series are non-normal and non-stationary, while they present negative skewness and positive kurtosis.[2]In order to overcome the non-stationarity issue, all series are transformed into daily series of rate returns using the following formula:

$$R_t = \ln\left(CP_t / CP_{t-1}\right) \tag{1}$$

where R_t is the rate of return and CP_t is the closing price (adjusted for dividends and stock splits) at time t. The descriptive statistics of the return series are shown in the following table:

Table 2: Summary statistics

	Ticker	UUP	DIA	FXB	VGK	USO	IAU
2010-2012	Mean	-0.00008	0.00038	-0.000003	0.00013	-0.00021	0.00055
	Standard deviation	0.0053	0.0105	0.0053	0.0185	0.0183	0.0111
	Skewness	0.2197	-0.4173	-0.1588	-0.2733	-0.3209	-0.5289
	Kurtosis	3.6456	6.3288	3.2348	5.9277	4.8466	5.7288
	Jarque-Bera (p value)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	ADF (p value)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
2013-2015	Mean	0.00022	0.00047	-0.00014	0.00017	-0.00147	-0.00062
	Standard deviation	0.0047	0.0078	0.0047	0.0097	0.0186	0.0109
	Skewness	-0.2642	-0.2714	-0.0862	-0.3376	-0.0481	-0.9357
	Kurtosis	5.0977	5.1789	4.7258	4.2069	5.7632	11.4728
	Jarque-Bera (p value)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
	ADF (p value)	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000

[1] The transaction costs for the three ETFs tracking their respective benchmarks do not exceed 0.5% per annum for medium size investors (see, for instance, www.interactive-brokers.com). Before the expansion of ETFs, traders had to pay a separate commission for each individual stock of an industry-specific portfolio. Now there are sector-specific ETFs, which allow traders to pay only one commission to buy or sell short an entire group of stocks.

[2] The Jarque-Bera statistics (1980) confirm their non-normality at the 99% confidence interval.

All returns series exhibit small skewness and high kurtosis. The Jarque-Bera statistic confirms that the six return series are non-normal at the 99% confidence level. The Augmented Dickey-Fuller (ADF) reports that the null hypothesis of a unit root is rejected at the 99% statistical level for all ETFs.

The proposed methodology and its benchmarks are applied on the task of forecasting and trading the one day ahead rate of return ($E(R_t)$) of the six ETFs. More specifically, the models' performance is evaluated through the four forecasting exercises presented in table 3.

Table 3: The total dataset

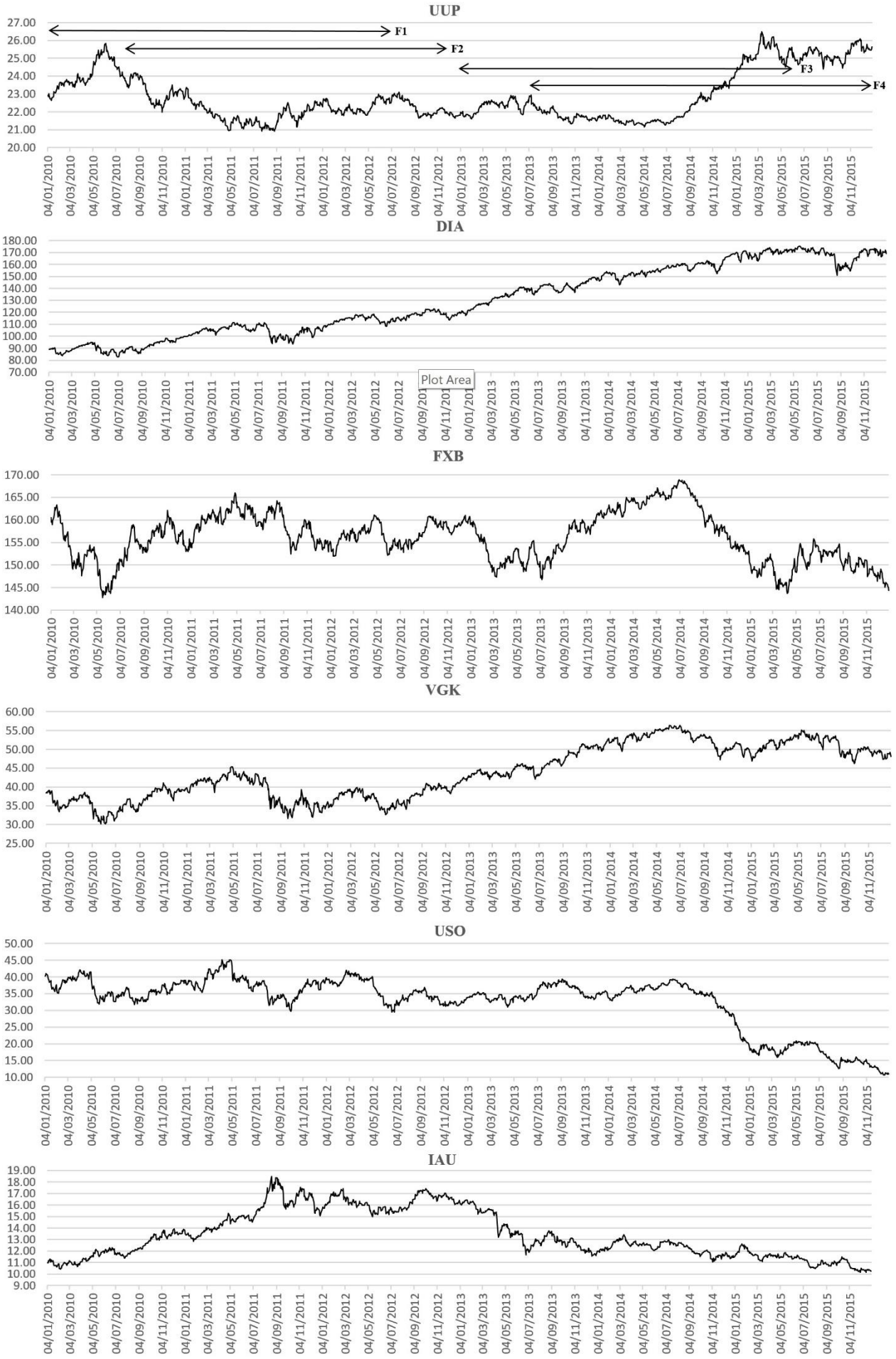
Note: The in-sample periods are the sum of the training and test datasets. The datasets of forecasting exercises F2 and F4 are simply formed by rolling the dataset of forecasting exercises F1 and F3 periods respectively by six months.

FORECASTING EXERCISE	PERIODS	TRADING DAYS	START DATE	END DATE
F1	Total Dataset	629	04/01/2010	29/06/2012
	Training Dataset	377	04/01/2010	30/06/2011
	Test Dataset	127	01/07/2011	30/12/2011
	Out-of-sample Dataset	125	03/01/2012	29/06/2012
F2	Total Dataset	630	01/07/2010	31/12/2012
	Training Dataset	380	01/07/2010	30/12/2011
	Test Dataset	125	03/01/2012	29/06/2012
	Out-of-sample Dataset	125	02/07/2012	31/12/2012
F3	Total Dataset	628	02/01/2013	30/06/2015
	Training Dataset	376	02/01/2013	30/06/2014
	Test Dataset	128	01/07/2014	31/12/2014
	Out-of-sample Dataset	124	02/01/2015	30/06/2015
F4	Total Dataset	632	01/07/2013	31/12/2015
	Training Dataset	380	01/07/2013	31/12/2014
	Test Dataset	124	02/01/2015	30/06/2015
	Out-of-sample Dataset	128	01/07/2015	31/12/2015

The four forecasting exercises cover the extended European sovereign debt crisis, including its peak (F1, F2) and its aftermath (F3, F4). The intuition behind the selection of the dataset is threefold. Firstly, the performance of all models will be examined under stressed market conditions. Secondly, this performance will be compared through different forecasting exercises aiming at observing the respective performances in the US and European markets. Lastly, detecting a connection between the empirical results with the AMH proposal will be attempted. Figure 1 presents the performance of the six ETFs during the four forecasting exercises.

Figure 1: The ETFs under study

Note: The four forecasting exercises are illustrated in the UUP graph. These forecasting exercises apply to all ETF series.



All models will be optimized in the in-sample and their forecasts will be evaluated in the out-of-sample.

3. Theoretical Framework

Support Vector Machines (SVMs) are learning machines utilizing the structural risk minimization principle to obtain good generalization on limited number of learning patterns (Wu and Liu, 2007). One class of SVM methods is the Support Vector Regression (SVR), introduced by Vapnik (1995), which is established as a robust technique for constructing data-driven and non-linear empirical regression models.

3.1 v-SVR

Considering the training data $\{(x_1,y_1), (x_2,y_2)\ldots, (x_n, y_n)\}$, where $x_i \in X \subseteq R, y_i \in Y \subseteq R, i = 1...n$ and n the total number of training samples, then the SVR function can be specified as:

$$f(x) = w^T \varphi(x) + b \tag{2}$$

$\varphi(x)$ is the non-linear function that maps the input data vector x into a feature space where the training data exhibit linearity (see figure 2c) while w and b are estimated by minimizing the regularized risk function:

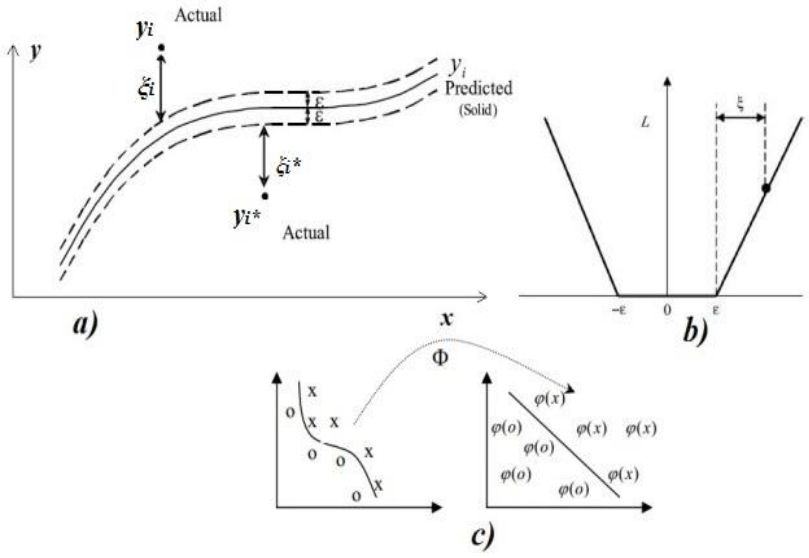
$$R(C) = C\frac{1}{n}\sum_{i=1}^{n} L_{\varepsilon}(y_i, f(x_i)) + \frac{1}{2}\|w\|^2 \tag{3}$$

The parameters C and ε are predefined by the practitioner, y_i is the actual value at time i and $f(x_i)$is the predicted value at the same period. The ε-sensitive loss L_{ε} function (see figure 2b) is defined as:

$$L_{\varepsilon}(y_i, f(x_i)) = \begin{cases} 0 \ \ if \ |y_i - f(x_i)| \leq \varepsilon \\ |y_i - f(x_i)| - \varepsilon \ \ if \ \ other \end{cases}, \varepsilon \geq 0 \tag{4}$$

Equation (4) identifies the predicted values that have at most ε deviations from the actual values y_i. The ε parameter defines a 'tube' (see figure 2a). The two variables, ξ_i and ξ_i^* represent the distance of the actual values from the upper and lower bound of the 'tube' respectively.

Figure 2: a) The ε-tube b) The plot of the ε-sensitive loss function c) The mapping procedure by φ(x)



The goal is to solve the following argument:

$$\text{Minimize } C\sum_{i=1}^{n}(\xi_i+\xi_i^*)+\frac{1}{2}\|w\|^2 \text{ subject to } \begin{Bmatrix}\xi_i\geq 0\\ \xi_i^*\geq 0\\ C>0\end{Bmatrix} \text{ and } \begin{Bmatrix}y_i-w^T\varphi(x_i)-b\leq+\varepsilon+\xi_i\\ w^T\varphi(x_i)+b-y_i\leq+\varepsilon+\xi_i^*\end{Bmatrix} \tag{5}$$

The quadratic optimization problem of equation (5) is transformed into a dual problem and its solution is based on the introduction of two Lagrange multipliers a_i, a_i^* and mapping with a kernel function $K(x_i, x)$:

$$f(x)=\sum_{i=1}^{n}(a_i-a_i^*)K(x_i,x)+b \text{ where } 0\leq a_i, a_i^*\leq C \tag{6}$$

The application of the kernel function transforms the original input space into one with more dimensions, where a linear decision border can be identified. In this study, the Gaussian Radial Basis Function (RBF) for all SVRs is applied. A RBF kernel is in general specified as:

$$K(x_i,x)=\exp(-\gamma\|x_i-x\|^2),\gamma>0 \tag{7}$$

where γ is the variance of the kernel function.

RBFs require only one parameter to be optimized (the γ) and provide good forecasting results in similar SVR applications (see amongst others Lu *et al.*, (2009), Yeh *et al.*, (2011) and Kao *et al.* (2013)).[3]

[3] In this study, we have also experimented with the Wavelet kernel (Zhang *et al.*, 2004) and the Mahalanobis kernel (Ruiz and Lopez-de-Teruel, 2001). However, their use did not provide better results than the ones obtained with the application of the RBF kernel. For the shake of space, these results are not presented but are available upon request.

Factor b in equation (6) is computed following the Karush-Kuhn-Tucker conditions. A detailed mathematical explanation of the solution can be found in Vapnik (1995). Support Vectors (SVs) are called the x_i's that lie closest to the ε margin, whereas non-SVs lie within the ε-tube. Increasing ε leads to more SVs' selection, whereas decreasing it results to more 'flat' estimates. The norm term $\|w\|^2$ characterizes the complexity (flatness) of the model and the term $\left\{\sum_{i=1}^{n}(\xi_i+\xi_i^*)\right\}$ is the training error, as specified by the slack variables. Consequently the introduction of parameter C satisfies the need to trade model complexity for training error and vice versa (Cherkassky and Ma, 2004).

The v-SVR algorithm can be used to make the optimization task easier, by encompassing the ε parameter in the optimization process and controls it with a new parameter $v \in (0,1)$. In v-SVR the optimization problem transforms to:

$$\text{Minimize } C\left(v\varepsilon+\frac{1}{n}\sum_{i=1}^{n}(\xi_i+\xi_i^*)\right)+\frac{1}{2}\|w\|^2 \text{ subject to } \left\{\begin{matrix}\xi_i \geq 0\\ \xi_i^* \geq 0\\ C>0\end{matrix}\right\} \text{ and } \left\{\begin{matrix}y_i - w^T\varphi(x_i) - b \leq +\varepsilon+\xi_i\\ w^T\varphi(x_i)+b-y_i \leq +\varepsilon+\xi_i^*\end{matrix}\right\} \quad (8)$$

The methodology remains the same as in ε-SVR and the solution takes a similar form:

$$f(x)=\sum_{i=1}^{n}(a_i-a_i^*)K(x_i,x)+b \quad \text{where} \quad 0 \leq a_i, a_i^* \leq \frac{C}{n} \quad (9)$$

Based on the 'v-trick', presented by Schölkopf *et al.* (1999), increasing ε leads to the proportional increase of the first term of $\left\{v\varepsilon+\frac{1}{n}\sum_{i=1}^{n}(\xi_i+\xi_i^*)\right\}$, while its second term decreases proportionally to the fraction of points outside the ε-tube. So v can be considered as the upper bound on the fraction of errors. On the other hand, decreasing ε leads again to a proportional change of the first term, but also the second term's change is proportional to the fraction of SVs. That This means ε will shrink as long as the fraction of SVs is smaller than v, therefore v is also the lower bound in the fraction of SVs. For a more detailed mathematical analysis of solutions see Vapnik (1995).

The superiority of vSVRs over the eSVR is well documented in the literature (see amongst others Cherkassky and Ma (2004) and Basak *et al.* (2007)). For this reason, the vSVR approach is followed in this study.

3.2. Locally Weighted Support Vector Regression

The locally weighted regression (LWR) is a memory-based procedure for fitting a regression surface to the data through multivariate smoothing. It is based on the assumption that the nearest to the predictor values are its best indicators. This is extremely beneficial in problems such as modelling financial trading

series, where some training points are more important than others and more recent observations have higher weight in predicting the future.

LWR can approximate an estimate g(x) of the regression surface for every value x in the dimensional space of the independent variables. Following the suggestions of Cleveland and Devlin (1988) each point of the neighbourhood is weighted according to its distance from point of interest x. The neighborhood is set by estimating the distances of q observations x_i from x, where $1 \leq q \leq n$ Those points that are close to x are assigned large weights, while those that are far have small weights. This confirms the local element of the method (Lee *et al.*, 2005). The idea of assigning weights to each point of the dataset could be expressed as:

$$\{(x_i, y_i, w_i)\}_{i=1}^{n}, x_i \in X \subseteq R, y_i \in Y \subseteq R, \ 0 \leq w_i \leq 1 \tag{10}$$

A quadratic function of the independent variables is fitted to the dependent variable using weighted least squares with these weights. In that way, g(x) is taken to be the value of this fitted function at x. A distance function in the space of the independent variables and a weight function to specify the neighborhood size are needed. The work of Cleveland and Devlin (1988) provides a detailed description on how to select these. The most common approach and the one followed in this study is to use the ratio q/n as a smoothness factor. The practitioner should interpret the smoothing factor rather than the q. The reason for that is that increasing the smoothing factor provides a smoother g(x) estimate. The selected weight function is the tricubic one specified below:

$$W(u) = \begin{Bmatrix} (1-u^3)^3, 0 \leq u \leq 1 \\ 0, otherwise \end{Bmatrix} \tag{11}$$

Based on these, the weight of each training data (x_i, y_i) is:

$$w_i(x_i) = W\left(\frac{\rho(x, x_i)}{d(x)}\right) \tag{12}$$

where ρ is the Euclidian distance and $d(x)$ is the Euclidian distance specifically from the q^{th}-nearest x_i to x. [4]

From equation (11) it is verified that $w_i \in [0,1]$. The weight has its maximum value when x_i is closest to x and its minimum for the q^{th}-nearest x_i to x.

[4] For instance, when the daily return for ETFs is desired and the expected return for the respective time series is roughly equal to zero, outliers of 5% gain per ticker could be ignored due to major structural change rather than routine behaviour of the time series. In this example x is the expected return target when modelling the deviation from this point, the outlier is x_i and w_i is the weight the Euclidean distance assigns to this observation.

Applying the principles of LWR to the SVR, we can achieve a Locally Weighted SVR (LSVR), where the parameter C is not constant, but locally adjusted as: $C_i = w_i * C$ (13)

In the case of ε-SVR, the quadratic optimization problem of equation (5) is transformed to:

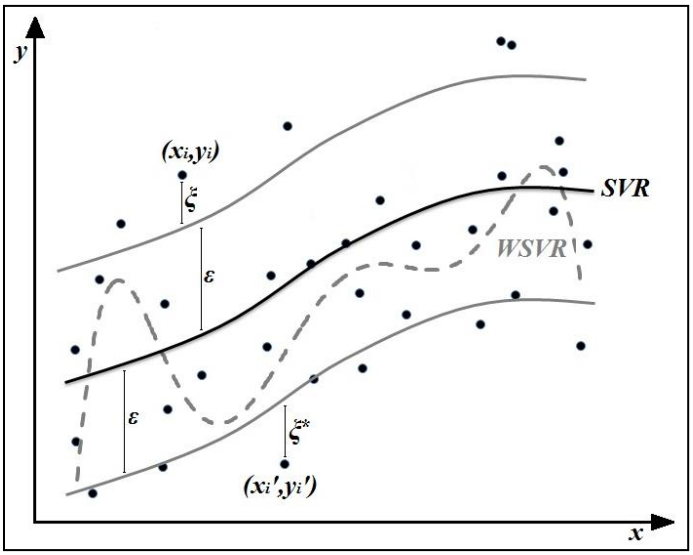
$$\text{Minimize } \sum_{i=1}^{n} C_i(\xi_i + \xi_i^*) + \frac{1}{2}\|w\|^2 \text{ subject to } \begin{Bmatrix} \xi_i \geq 0 \\ \xi_i^* \geq 0 \\ C_i > 0 \end{Bmatrix} \text{ and } \begin{Bmatrix} y_i - w^T\varphi(x_i) - b \leq +\varepsilon + \xi_i \\ w^T\varphi(x_i) + b - y_i \leq +\varepsilon + \xi_i^* \end{Bmatrix} \quad (14)$$

Regarding v-SVR, the quadratic optimization problem of equation (8) becomes the following:

$$\text{Minimize } C_i\left(v\varepsilon + \frac{1}{n}\sum_{i=1}^{n}(\xi_i + \xi_i^*)\right) + \frac{1}{2}\|w\|^2 \text{ subject to } \begin{Bmatrix} \xi_i \geq 0 \\ \xi_i^* \geq 0 \\ C_i > 0 \end{Bmatrix} \text{ and } \begin{Bmatrix} y_i - w^T\varphi(x_i) - b \leq +\varepsilon + \xi_i \\ w^T\varphi(x_i) + b - y_i \leq +\varepsilon + \xi_i^* \end{Bmatrix} \quad (15)$$

The conceptual advantage of the LSVR against the traditional SVR is based on the locally adjusted C_i. The weighted C_i provides better balance between training error and model complexity, by penalizing more (small weight) big slack variables (see figure 3).

Figure 3: the ε-tube (grey lines), the f(x) curves of SVR (black solid line) and WSVR (grey dashed line)



The traditional SVR attempts to track all training data with a specific model complexity through a constant C. Conceptually, this suggests that that the size of ξ_i, ξ_i^* does not fluctuate much. The LSVR, on the other hand, highly penalizes the errors near x in an attempt to increase predictability. The SVR and LSVR models are based on optimizing an objective function that tunes the parameters. In the process of optimization/training the kernel as of equations (14) and (15), potential misclassifications are penalized. The penalty term can be constant as C or it can be adaptive to some measures as C_i. In the proposed algorithm, the penalty term is dependent on the associated weight to the point and the weight

is defined based on Euclidean distance (equation 12). In this manner, when the previous points (in-sample) are used to train the kernel in LSVR, their misclassification grade is penalized by the Euclidean distance function as of equation (11). LSVR's predictive performance is increasing gradually, as the shape of the weight function is becoming sharper (Lee *et al.*, 2005).[5] This is a clear point of superiority of LSVR over the non-locally optimized method. Nonetheless, it should be clarified that the weighting process does not solve 'over-fitting' issues that usually impede the success of SVR applications.

4. Methodology

This section summarizes the proposed methodology applied in the study. Initially, the KH algorithm is described along with the way it is combined with LSVR. Next, we introduce a novel extension of the KH-LSVR. Finally, the SVR input and model benchmark selection is presented.

4.1 Locally Weighted Krill Herd Support Vector Regression (KH-LSVR)

As discussed earlier, LSVR provides better balance between the training error and model complexity, which is crucial for the success of the SVR method. However, this does not provide a safety net to affront the major challenge of 'over-fitting'. The most common SVR optimization approaches, the cross-validation and grid search, are data and task biased (Zhang *et al.*, 1999).

For that reason, we propose a hybrid Krill Herd – Support Vector Regression that embodies a KH algorithm for optimal parameter selection to the LSVR process, as shown in section 3.2. The KH algorithm, as presented by Gandomi and Alavi (2012), is an innovative metaheuristic optimization technique that simulates the herding behavior of krill individuals. The intuition of the analysis is the mean-reversion effect of predators' attacks on the herd of krills. Such attacks result in the reduction of the krill density of the herd. After the attack, the herd must increase its density by sensing nearby krill but without deviating much from the optimal path to reach food.

Based on the above, Gandomi and Alavi (2012) propose that the position (*P*) of each krill in the search space is influenced by the movement induced by other krill (*M*), the foraging action (*F*) and the random diffusion (*RD*). All these motions can be summarized in one Lagrangian formulation for every krill *j*:

$$\frac{dP_j}{dt} = M_j + F_j + RD_j \tag{16}$$

The new movement motion M^{t+1} of each krill *j* is calculated as:

[5] The weight of every point for a traditional regression model is 1/n, meaning that the assigned weight is similar for every point. In LWR the importance/weight increases continuously, once we move from outliers to more central points. Plotting the weight functions for both cases, it is obvious that in the traditional regression the function is a horizontal line, whereas in LWR the function is has a bell-shape around the central point. Once the importance of distance is increased through higher orders of power, the LWR function plot becomes narrower or in other words sharper.

$$\left\{\begin{array}{l} M_j^{t+1} = M^{\max} eff_j + k_M \mathrm{M}^t \\ eff_j = eff_j^{loc} + eff_j^{t\arg} \end{array}\right\} \quad (17)$$

Where:

- M^{max}: the maximum induced speed
- $k_M \in [0,1]$: the inertia weight of the motion
- M_j^t: the previous movement motion
- eff_j: the direction of the motion
- $eff_j^{loc}, eff_j^{t\arg}$:the local effect by neighbor krill and the target direction effect by the best individual krill.

Gandomi and Alavi (2012) suggest that the local search of the algorithm is based on an attractive/repulsive tendency between individual krills. The neighbor krills are identified through a sensing distance from the j^{th} one:

$$d_{s,j} = (1/\mathrm{N_k}) \sum_{j'=1}^{N_k} \left\| P_j - P_{j'} \right\| \quad (18)$$

where N_k is the number of krill individuals.

The new foraging motion F^{t+1} of every krill j is also calculated on the basis of two factors, namely the food location and its previous experience in locating a correct food position:

$$\left\{\begin{array}{l} F_j^{t+1} = V_F \, floc_j + k_F F_j^t \\ floc_j = floc_j^{food} + floc_j^{best} \end{array}\right\} \quad (19)$$

where:

- V_F : the foraging speed[6]
- $k_F \in [0,1]$: the inertia weight of the motion
- F_j^t: the previous foraging motion
- $floc_j$: the location of the food
- $floc_j^{food}$, $floc_j^{best}$: the food attractive and the effect from the best food-locating j^{th} krill so far

The food attraction is defined to provide global optima for the krill swarm. The third motion RD of krills is calculated as a maximum diffusion speed RD^{max} and a random directional vector δ with values between -1 and 1. In other words: $RD = RD^{\max} \delta$ (20)

The equations (17) and (19) suggest the future krill motion towards the optimal position by performing two parallel local and global strategies, something that makes the KH algorithm very robust. Krill

[6] The maximum induced speed of equation (17) and the foraging speed of equation (19) are set to 0.01 and 0.02 ms^{-1} respectively, as Gandomi and Alavi (2012) suggest.

continue their local search (equation (17)) until the herd density increases. When that happens, more and more krill orientate to food (equation (19)) rather than the nearby krill. These two strategies provide the fitness values for several effective factors that induce an attractive/repulsive motion response to each krill. The equation (20) performs a random search in the proposed search space, diffusing any potential biased motion responses to the herd (either towards food locations or neighboring sensed krill). For more details on the approximation of these values, refer to the extensive mathematical steps of Gandomi and Alavi (2012). The position *Pj* of each krill at time $t+\Delta t$ is given as:

$$\left\{ \begin{array}{l} P_j(t+\Delta t) = P_j(t) + \Delta t \dfrac{dP_j}{dt} \\ \Delta t = Z_{cr} \sum_{r=1}^{NP} \left(UpB_r - LowB_r \right) \end{array} \right\} \tag{21}$$

where:

- $Z_{cr} \in [0,2]$: constant number
- NP: the number of parameters optimized (in our case NP=3)
- $UpB_r, LowB_r$: the upper and lower bounds of the parameters

The Δt practically is the only parameter that needs fine tuning. This is the striking advantage of the method compared to other more complicated metaheuristics approaches.

In the KH-LSVR optimization, the practitioner needs to predefine three parameters. The range of the bounds of each parameter defines the potential three-dimensional search space. The Z_{cr} is set at values lower than 1, because it allows careful search of the space by the krill individuals. Krill behavior suggests that herd individuals at an initial point (predator attack) tend to focus on exploration of the search space and then its exploitation. For that reason, k_M and k_F are initially set high (0.9). These parameters are linearly decreased to 0.1 at the end to encourage exploitation (Gandomi and Alavi, 2012). Next two genetic reproduction mechanisms (mutation and crossover) are implemented in order to further improve the performance of the krill positions.

The KH algorithm is optimized based on a fitness function. In similar applications of metaheuristics, the practitioners set as fitness function a simple statistical measure such as the MSE or the RMSE. However, in financial forecasting applications the trading performance is of the outmost importance. In this research, the algorithm is trained through three different fitness functions. This exercise should improve the forecasting performance of our models. The choice of the fitness function is based on the performance of the models in the in-sample. The first function aims to minimize the RMSE (equation (22)). The second one aims to maximize the annualized return (equation (23)) while the third (equation (24)) attempts to provide a balance between statistical accuracy and trading efficiency.

$$Fitness_1 = 1/(1+RMSE) \quad (22)$$

$$Fitness_2 = Annualized\ Return \quad (23)$$

$$Fitness_3 = Annualized\ Return - 10 * RMSE \quad (24)$$

The aim of the algorithm is to maximize equations (22), (23) and (24). The KH algorithms are trained in the training sub-period and their performance is evaluated in the test sub-period. The fitness is evaluated in terms of Annualized Return. The function that performs better will be applied in the out-of-sample. The outcome of this process is presented in appendix B. We note that $Fitness_3$ seems to dominate the selection process. Nevertheless, there are several cases that the other two functions are selected. These results outline the importance of experimenting on the metaheuristics fitness function in the in-sample. Following a priori selections can deter the performance of the metaheuristics algorithm. The flowchart of the KH-LSVR is presented in figure 4(i).

4.2. Reverse-adaptive KH-LSVR (RKH-LSVR)

The success of the original KH algorithm is well documented in the metaheuristics literature. Nonetheless, KH can trap into local optima which constraints the efficiency of the implemented global search. Additionally, the KH depends completely on a random search process and as such there is no guarantee for a fast convergence to optimal solutions (Bolaji *et al.*, 2016). In order to cope with these issues, we propose the novel Reverse-adaptive KH algorithm (RKH) for the parameterization of the LSVR. The RKH improves the original KH process by incorporating opposition-based optimization (OBL) and a two-stage operator to reach the final optimal solutions.

4.2.1 Reverse Krill Improvement

The first enhancement of the KH algorithm is derived by allowing reverse points in the selection of the population. OBL has gained significant attention in metaheuristic optimization algorithms (Hu *et al.*, 2014 ; Huang *et al.*, 2016). The basic principle of OBL is that some variables discard their current values for their opposite ones, based on their comparison of the goodness of fit.

In the case of KH, let $P_j = (p_1, p_2, ..., p_D)$ the candidate solution in a *D*-dimensional search space. Based on OBL, the reverse point is defined as $P_j^* = (p_1^*, p_2^*, ..., p_D^*)$, where $p_{\bar{\pi}}^* = \zeta_{\bar{\pi}} + \overline{\zeta}_{\bar{\pi}} - p_{\bar{\pi}}$, $p_{\bar{\pi}} \in [\zeta_{\bar{\pi}}, \overline{\zeta}_{\bar{\pi}}]$ and $\bar{\pi} \in 1, 2, ...D$. Obtaining the reverse position point is crucial, as we can now compare the fitness between the original P_j and its opposite P_j^*. Such comparison can increase the diversity of the krill population through a simple comparison of their density function values $\bar{F}$:

$$\left\{\text{if } \bar{F}(P_j^*) < \bar{F}(P_j)\text{, then population is updated with } P_j^*\text{; otherwise keep } P_j \text{ in the population}\right\} \quad (25)$$

RKH algorithm computes and evaluates P_j and P_j^* simultaneously in order to converge to the best population. Although it maintains its ability to generate random candidate solutions for the population as KH, this happens only partially. Initially, the first half of the population (POP) is randomly formed as in the traditional KH, but the other half (POP_{rev}) is generated through rule (25) and (POP). The update of the krill positions of the initial POP is done based on the process described in 4.1, while the POP_{rev} krill update is always subject to rule (25) and the relevant POP at the time. Once the optimal solutions are sorted in each population, the final population is formed by merging POP and $POP_{rev.}$ This approach vastly increases the search ability of krill by allowing smoother and quicker movement towards the best solutions. It should be noted that with RKH the population size remains unchanged throughout the optimization, while solutions are ranked according to fitness. Based on these, identifying the best solution in each generation is easier.

4.2.2 Two-stage operator Improvement

The RKH process is further enhanced in terms of optimal convergence by applying the Cauchy distribution and 'krill-grip' for mutation and crossover purposes respectively. This combination comprises a hybrid two-stage operator (CPO) that decreases vastly the probability of KH getting trapped in local optima. It also further improves the search ability of the algorithm, since the global best individual (identified easier through OBL as in section 4.2.1) of each generation is passed on to the next generations along with its neighbours. As a consequence, convergence speed gradually increases. The CPO follows the reverse population update described in the previous section.

The Cauchy mutation operator in the RKH context is applied to the best candidate solution of each generation through a weight and position update function. The weight function is defined as:

$$\bar{W}(\bar{j}) = \left(\sum_{j'=1}^{N_K} P_{j\,\bar{j}} \right) / N_K \tag{26}$$

where $P_{j\,\bar{j}}$ is the $\bar{j}$ th position vector of the jth krill, N_K is the population size and $\bar{W}(\bar{j}) \in [-1,1]$.

The position update vectors are computed as $P'(\bar{j}) = P(\bar{j}) + \bar{W}(\bar{j}) * \bar{C}$ (27)

where $\bar{C}$ is randomly drawn from a Cauchy distribution with τ=1[7].

Based on these weights, the neighbourhood of each best solution is selected. Although the previous process increases the convergence speed, it needs to be ensured that the algorithm maintains the balance between the search space exploration and exploitation. This balance is distorted when position values become very large for distant krill from the best solutions. For that reason, we also introduce position

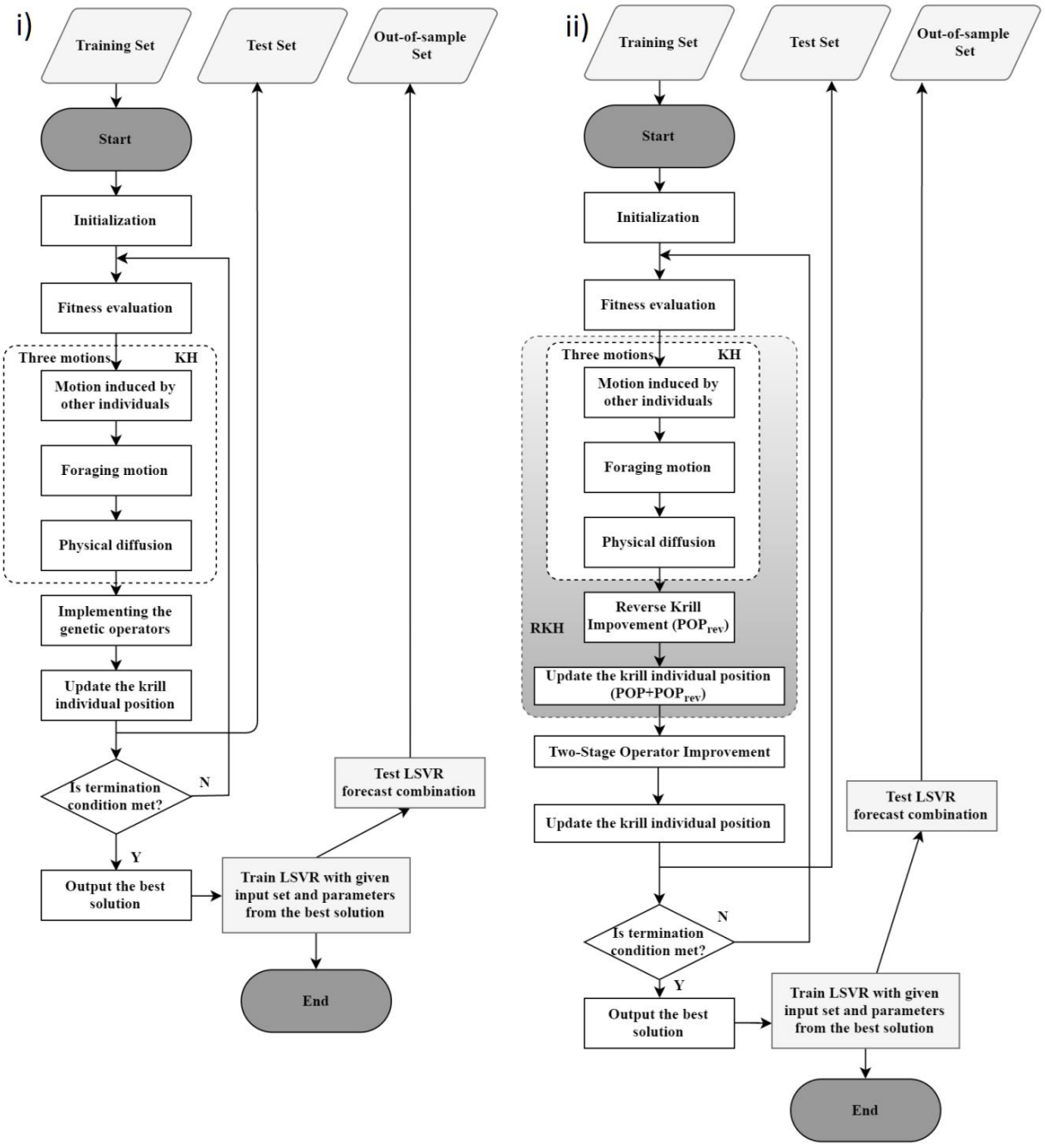
[7] From an algebra point of view, the one-dimensional Cauchy density function is given by $f(\chi) = t/(\pi * \tau^2 + \tau^2), x \in R, \tau > 0$. The Cauchy distributed function is calculated by $F_\tau(\chi) = 0.5 + (\pi^{-1} \arctan(\chi/\tau))$.

constraints to krill movement in order to 'grip' krill that distance themselves too much from the best candidate. This second stage of the CPO simply creates a maximum bound for the global search by clamping the positions of krill. To put it simply, once a krill j in the $\bar{j}$ th dimension searches too far from the best ones, then its position is set to the maximum allowed position $P_{j\bar{j}}\max$. Given these, the update of the position based on clamping is:

$$P_{j\bar{j}}(t+1)=\begin{cases} P_{j\bar{j}}'(t+1), \textit{if } P_{j\bar{j}}(t+1) < P_{j\bar{j}}\max \\ P_{j\bar{j}}\max, \textit{otherwise} \end{cases} \tag{28}$$

These two steps complete the CPO operator. Once the CPO is applied to the merged population (POP, POP_{rev}), the best solution is found and evaluated through the same fitness functions as in the KH-LSVR (equations (22), (23), (24)). If not, the iteration starts again from the usual three-motion KH process. Similarly to the case of KH-LSVR, $Fitness_3$ dominates the selection process. This is shown along with all the technical characteristics of the KH algorithms in appendix B. The flowchart of the proposed novel RKH-LSVR is shown in figure 4(ii) below.

Figure 4: The KH-LSVR (i) and RKH-LSVR (ii) flowcharts



4.3 Input Selection

In order to select the SVR inputs, a set of potential linear and non-linear predictors for the in-sample is generated. The linear pool includes Simple Moving Averages (SMA), Exponential Moving Averages (EMA), Autoregressive terms (AR), Autoregressive Moving Average (ARMA) models, Rate of Change Indicators (ROC), and a Pivot Point Indicator (PPI). The non-linear Smooth Transition Autoregressive Models (STAR), Nearest Neighbors Algorithms (*k*-NN), a Multi-Layer Perceptron (MLP), a Recurrent Neural Network (RNN), a Higher Order Neural Network (HONN), a Psi-Sigma Neural Network (PSN), Adaptive RBF and PSO Neural Network (ARBF-PSO), Genetic Programming (GP) and Gene Expression Programming (GEP) complete the input pool. These predictors create a pool of three hundred and forty eight individual predictors in total for each forecasting exercise and each ETF series. All predictors have been successfully applied in financial forecasting applications. The proposed models

will combine the best predictors in order to generate superior out-of-sample performance. Combining individual linear and non-linear forecasts to improve forecast accuracy is common practice among practitioners and researchers (Timmermann, 2006; Park and Irwin, 2007). Forecast combinations generate more robust signals and in trading applications offer the benefits of model diversification. A short summary of the models that consist the input pool is provided in Appendix A. The dimensions of the potential input vector are large. In order to cope with the dimensionality issue, we perform a simple PCA analysis (Jolliffe, 2002). PCA allows us to discard highly correlated variables, while preserving those ones that account for the 95% variance of the dataset. The outcome of this analysis provides the principal components that are going to be used as final input sets to the SVR models. These sets are presented in table 4.

Table 4: The SVR sets of inputs

	UUP	DIA	FXB	VGK	USO	IAU
F1	AR(2), EMA(4), MLP, RNN, HONN, PSN	EMA(6), ARMA(2,3), MLP, HONN, GP, GEP, ARBF-PSO	MLP, RNN, HONN, PSN, ARBF-PSO	MLP, RNN, HONN, PSN, ARBF-PSO	LSTAR(4), ARMA(1,4), PSN, GP, GEP, ARBF-PSO	ARMA(2,4), EMA(4), MLP, RNN, HONN, GEP
F2	ARMA(1,6), ROC(3), MLP, RNN, ARBF-PSO	SMA(5), ESTAR(5), MLP, GP, GEP, ARBF-PSO	AR(5), SMA(3), MLP, RNN, HONN, PSN, GP	SMA(8), *k*-NN, MLP, RNN, PSN, GEP, ARBF-PSO	AR(5), ARMA(1,6), ARMA(3, 3), ESTAR(6), MLP HONN, PSN, GP, ARBF-PSO	AR(3), EMA(5), MLP, RNN, HONN, PSN, GP, GEP
F3	AR(2), SMA(4), EMA(3), MLP, RNN, HONN, PSN, GP, GEP	ROC(4), *k*-NN, MLP, RNN, HONN, PSN, ARBF-PSO	MLP, RNN, PSN, ARBF-PSO	AR(3), ARMA(4,6), HONN, GP, GEP, ARBF-PSO	MLP, RNN, HONN, PSN, GP, GEP, ARBF-PSO	AR(3), ARMA(4,6), ARMA(4, 8), HONN, PSN, GP, GEP, ARBF-PSO
F4	AR(4), ARMA(3,6), PSN, GP, GEP, ARBF-PSO	MLP, RNN, PSN, ARBF-PSO	AR(1), ARMA(1,2), MLP, PSN, GP, GEP, ARBF-PSO	SMA(7), ARMA(3, 8), MLP, RNN, PSN, ARBF-PSO	SMA(4), ESTAR(3), MLP, RNN, PSN, ARBF-PSO	PSN, GP, GEP, ARBF-PSO

The above table shows that the PCA analysis decreases substantially the dimensions of the input pool. The non-linear models dominate the selected input sets.

4.4 Benchmark Models

The efficiency of the RKH-LSVR is evaluated in this study through several benchmarks. SVR practitioners face a constant tackle, the optimal selection of the model's parameters (C, ε or v and kernel function parameter). The most well documented parameterization technique is the grid search, while GAs[8] dominates the metaheuristics literature on SVR optimization. Based on this fact, a SVR optimized through grid search (vSVR) and a SVR optimized through GA (GA-vSVR) act as benchmarks to the proposed model. In order to examine the benefits of RKH over the KH algorithm, a LSVR optimized through KH (KH-LSVR) will benchmark the RKH-LSVR performance. All SVR procedures use a set of inputs selected from a large pool of linear and non-linear potential predictors (see table 4). In order to study the utility of the SVR algorithms, the best models from this pool in terms of in-sample statistical

[8] For more details on the GA algorithm see Holland (1975). Its characteristics are presented in Appendix B. Similarly, as the KH, the GA algorithm is centered around a fitness function, The fitness function selection process that is presented on section 4.1 is also applied to the GA-*v*SVR model. These results are also presented in Appendix B.

accuracy (RMSE) and trading performance (annualized return) in the in-sample will also act as benchmarks. These models are presented in table 5.

Table 5: The best individual predictors (linear and non-linear benchmarks)

		UUP	DIA	FXB	VGK	USO	IAU
RMSE	F1	EMA(4)	GP	PSN	PSN	LSTAR(4)	ARMA(2,4)
	F2	ARBF-PSO	ARBF-PSO	PSN	ARBF-PSO	ESTAR(6)	GEP
	F3	EMA(3)	PSN	ARBF-PSO	ARMA(4,6)	PSN	GEP
	F4	PSN	PSN	MLP	RNN	SMA(4)	MLP
Annualized Return	F1	PSN	ARBF-PSO	PSN	ARBF-PSO	ARBF-PSO	GEP
	F2	ARBF-PSO	ESTAR(5)	GP	GEP	GP	HONN
	F3	GEP	PSN	ARBF-PSO	ARBF-PSO	PSN	GP
	F4	ARBF-PSO	RNN	PSN	PSN	ARBF-PSO	PSN

A simple random walk (RW) with no trend will also act as naïve benchmark.

5. Statistical Evaluation

This section provides the out-of-sample statistical performance of all models applied. In 5.1 the statistical accuracy of the proposed models is presented while in 5.2 the genuineness of the forecasts is evaluated through the Giacomini and White (GW) (2006) test.

5.1 Statistical Accuracy

The statistical accuracy of the obtained forecasts is evaluated through the RMSE (see appendix C), the Pesaran-Timmermann (PT) (1992) test and the Diebold-Mariano (DM) (1995) test. The PT test is used to examine whether the directional movements of the real and forecast values are in step with one another. The null hypothesis is that the model under study has no power on forecasting the relevant ETF return series. The DM test checks the null hypothesis of equal predictive accuracy between our models' forecasts[9]. For more details on the PT and the DM test see Pesaran and Timmermann (1992) and Diebold and Mariano (1995) respectively. The out-of-sample results are summarized in table 6.

Table 6: Out-of-sample Statistical Performance

[9] In our exercise we apply the DM test to couples of forecasts (RKH-LSVR vs. another forecasting model). A rejection of the null hypothesis suggests that the first forecast (the RKH-LSVR) is more accurate.

*Note: The table reports the RMSE values of each forecast while the PT statistics are in the parenthesis. ***, ** and * denotes that the DM null hypothesis is rejected at the 1%, 5% and 10% significance level respectively. $Best_1$ and $Best_2$ refer to the best individual predictors in terms of statistical and trading performance respectively in the in-sample period (as outlined in table 5).*

	Market	ETF	RW	$Best_1$	$Best_2$	*v*SVR	GA-*v*SVR	KH-LSVR	RKH-LSVR
F1	US Currency	UUP	0.0073 (5.15)***	0.0070 (6.90)***	0.0071 (6.25)***	0.0068 (7.34)***	0.0067 (7.57)***	0.0063 (7.88)***	0.0059 (8.39)
	US Stock	DIA	0.0068 (5.95)***	0.0061 (6.87)***	0.0063 (6.74)***	0.0060 (7.52)***	0.0058 (8.17)***	0.0057 (8.67)***	0.0055 (9.05)
	EU Currency	FXB	0.0075 (4.18)***	0.0072 (5.97)***	0.0073 (5.05)***	0.0070 (6.54)***	0.0069 (7.29)***	0.0065 (7.87)**	0.0062 (8.98)
	EU Stock	VGK	0.0071 (5.45)***	0.0067 (6.04)***	0.0069 (6.72)***	0.0065 (6.15)***	0.0063 (7.43)***	0.0062 (8.04)***	0.0055 (8.72)
	Commodity: Oil	USO	0.0066 (6.31)***	0.0058 (7.55)***	0.0060 (7.08)***	0.0056 (7.68)***	0.0054 (8.40)***	0.0053 (9.15)***	0.0051 (9.50)
	Commodity: Gold	IAU	0.0064 (7.12)***	0.0060 (7.88)***	0.0062 (7.45)***	0.0056 (8.63)***	0.0052 (9.28)***	0.0050 (9.45)***	0.0047 (10.36)
F2	US Currency	UUP	0.0076 (4.54)***	0.0072 (5.65)***	0.0074 (5.07)***	0.0070 (6.28)***	0.0068 (7.05)***	0.0065 (7.65)**	0.0062 (8.53)
	US Stock	DIA	0.0070 (5.31)***	0.0066 (6.97)***	0.0065 (6.41)***	0.0062 (7.65)***	0.0060 (8.75)***	0.0059 (8.95)***	0.0053 (9.11)
	EU Currency	FXB	0.0078 (3.34)***	0.0075 (5.23)***	0.0077 (4.15)***	0.0074 (5.98)***	0.0072 (6.47)***	0.0069 (7.18)**	0.0065 (8.04)
	EU Stock	VGK	0.0074 (4.38)***	0.0070 (5.37)***	0.0072 (5.79)***	0.0069 (6.73)***	0.0066 (7.41)***	0.0063 (7.56)***	0.0060 (8.35)
	Commodity: Oil	USO	0.0068 (5.15)***	0.0062 (6.23)***	0.0064 (5.87)***	0.0059 (7.88)***	0.0056 (8.47)***	0.0055 (8.86)***	0.0053 (9.35)*
	Commodity: Gold	IAU	0.0066 (5.44)***	0.0061 (6.63)***	0.0063 (6.22)***	0.0058 (7.54)***	0.0054 (8.36)***	0.0052 (9.08)***	0.0050 (10.07)
F3	US Currency	UUP	0.0071 (5.15)***	0.0068 (6.80)***	0.0069 (6.35)***	0.0068 (7.45)***	0.0066 (8.20)***	0.0062 (8.88)***	0.0058 (9.58)
	US Stock	DIA	0.0066 (5.99)***	0.0060 (7.02)***	0.0061 (7.55)***	0.0058 (7.80)***	0.0057 (8.35)***	0.0055 (9.15)**	0.0051 (9.64)
	EU Currency	FXB	0.0074 (4.50)***	0.0070 (6.58)***	0.0072 (5.74)***	0.0069 (6.90)***	0.0067 (7.58)***	0.0063 (8.41)***	0.0060 (9.24)
	EU Stock	VGK	0.0069 (5.87)***	0.0066 (6.34)***	0.0067 (6.81)***	0.0064 (7.36)***	0.0063 (7.97)***	0.0060 (8.75)**	0.0054 (9.27)
	Commodity: Oil	USO	0.0064 (6.25)***	0.0056 (7.12)***	0.0058 (6.86)***	0.0053 (8.07)***	0.0051 (8.80)***	0.0049 (9.04)***	0.0046 (9.94)
	Commodity: Gold	IAU	0.0061 (6.80)***	0.0056 (7.95)***	0.0058 (7.40)***	0.0052 (8.36)***	0.0050 (8.99)***	0.0047 (9.48)***	0.0043 (10.77)
F4	US Currency	UUP	0.0072 (5.13)***	0.0068 (7.15)***	0.0070 (6.87)***	0.0067 (7.35)***	0.0065 (8.41)***	0.0061 (9.25)***	0.0056 (9.68)
	US Stock	DIA	0.0063 (5.54)***	0.0054 (7.42)***	0.0058 (6.35)***	0.0052 (8.26)***	0.0050 (8.99)***	0.0048 (9.85)***	0.0044 (10.25)
	EU Currency	FXB	0.0073 (4.90)***	0.0069 (6.85)***	0.0070 (6.22)***	0.0068 (7.05)***	0.0067 (8.07)***	0.0062 (9.05)***	0.0058 (9.41)
	EU Stock	VGK	0.0069 (4.42)***	0.0065 (5.57)***	0.0067 (5.79)***	0.0065 (6.73)***	0.0063 (6.99)***	0.0059 (8.57)***	0.0053 (9.05)
	Commodity: Oil	USO	0.0062 (5.96)***	0.0054 (7.78)***	0.0056 (6.93)***	0.0050 (8.66)***	0.0047 (9.10)***	0.0045 (9.80)***	0.0042 (10.32)
	Commodity: Gold	IAU	0.0058 (6.30)***	0.0055 (8.14)***	0.0056 (7.05)***	0.0050 (8.50)***	0.0047 (9.33)***	0.0044 (10.41)***	0.0041 (11.05)

The above table provides some interesting results. The proposed RKH-LSVR outperforms the original KH counterpart across different forecasting exercises and ETFs. Thus, the proposed improvements to the KH algorithm are indeed beneficial to the forecasting accuracy of the model. Additionally, the GA-*v*SVR is consistently superior to the traditional SVR, but it does not manage to match the forecasting efficiency of the two KH counterparts. The superiority of the SVR and hybrid SVR models is confirmed, since in all cases the best individual predictors (linear and non-linear) are inferior in terms of accuracy. the This suggests that using a large pool of predictors to extract optimal inputs based on the previous described PCA process leads to increased forecasting accuracy. As expected, the best individual

predictors in terms of statistical performance in-sample ($Best_1$) remain superior in the out-of-sample compared to $Best_2$. The DM test further demonstrates the superiority of the RKH-LSVR forecasts in all ETFs and periods under study. The significant PT statistics reveal that all models are capable of capturing the directional movements of the six ETF return series. The statistical accuracy of the forecasts deteriorates during F1 and F2 periods. In particular, the worst out-of-sample results are obtained in F2. Finally, it is also evident that forecast errors appear the smallest in the analysis of commodities and largest in the cases of currencies.

5.2. Giacomini-White test

The previous statistical results are further authenticated by computing the unconditional GW test for the out-of-sample predictive ability testing and forecast selection. The null hypothesis of the GW test is the equivalence in forecasting accuracy between two forecasting models. The sign of the test statistic specifies the superior model according to its forecasting performance. A positive realization of the GW test statistic indicates that the second model is more accurate than the first one whereas a negative specifies the opposite. The test is calculated based on the MSE loss function. The outcomes of the GW test are presented in table 7 below.

Table 7: The Giacomini-White test for all out-of-sample periods.

*Note: The table displays the p-values of the statistic under the null hypothesis that the column model shows equivalent performance compared with RKH-LSVR for every ETF separately. *** at the 1% significance level respectively. $Best_1$ and $Best_2$ refer to the best individual predictors in terms of statistical and trading performance respectively in the in-sample period (as outlined in table 5).*

	ETF	Models	RW	$Best_1$	$Best_2$	*v*SVR	GA-*v*SVR	KH-LSVR
F1	UUP	RKH-LSVR	0.000***	0.000***	0.001***	0.000***	0.002***	0.006***
	DIA	RKH-LSVR	0.000***	0.000***	0.000***	0.001***	0.001***	0.005***
	FXB	RKH-LSVR	0.001***	0.003***	0.003***	0.003***	0.004***	0.012**
	VGK	RKH-LSVR	0.000***	0.000***	0.002***	0.001***	0.007***	0.007***
	USO	RKH-LSVR	0.002***	0.002***	0.001***	0.000***	0.005***	0.008***
	IAU	RKH-LSVR	0.000***	0.000***	0.000***	0.004***	0.002***	0.003***
F2	UUP	RKH-LSVR	0.000***	0.000***	0.000***	0.001***	0.008***	0.020**
	DIA	RKH-LSVR	0.000***	0.000***	0.000***	0.000***	0.007***	0.006***
	FXB	RKH-LSVR	0.001***	0.001***	0.003***	0.005***	0.004***	0.015**
	VGK	RKH-LSVR	0.000***	0.000***	0.000***	0.001***	0.006***	0.008***
	USO	RKH-LSVR	0.000***	0.000***	0.000***	0.003***	0.007***	0.007***
	IAU	RKH-LSVR	0.001***	0.000***	0.000***	0.001***	0.003***	0.004***
F3	UUP	RKH-LSVR	0.000***	0.000***	0.000***	0.005***	0.009***	0.010**
	DIA	RKH-LSVR	0.000***	0.000***	0.000***	0.001***	0.005***	0.006***
	FXB	RKH-LSVR	0.000***	0.000***	0.000***	0.006***	0.008***	0.008***
	VGK	RKH-LSVR	0.000***	0.000***	0.000***	0.001***	0.003***	0.005***
	USO	RKH-LSVR	0.003***	0.001***	0.004***	0.001***	0.002***	0.004***
	IAU	RKH-LSVR	0.000***	0.000***	0.000***	0.006***	0.006***	0.005***
F4	UUP	RKH-LSVR	0.000***	0.000***	0.000***	0.002***	0.005***	0.007***
	DIA	RKH-LSVR	0.003***	0.000***	0.002***	0.004***	0.003***	0.012**
	FXB	RKH-LSVR	0.000***	0.000***	0.000***	0.001***	0.003***	0.003***
	VGK	RKH-LSVR	0.000***	0.000***	0.000***	0.004***	0.002***	0.005***
	USO	RKH-LSVR	0.000***	0.000***	0.000***	0.002***	0.005***	0.005***
	IAU	RKH-LSVR	0.000***	0.000***	0.000***	0.001***	0.003***	0.006***

The above results further validate the statistical findings of the previous section. RKH-LSVR is found to be statistically superior to its benchmarks, since the null hypothesis of the GW test is rejected in all cases at 1% significance level. Here it should be noted, that the predictive ability tests can be

distinguished as conditional and unconditional types. The unconditional type attempts to answer which model is performing better in forecasting on average. On the other hand, the conditional type tries to evaluate which model can predict better in the future over unseen inputs. DM and GW are two distinct tests for unconditional predictive ability (Clark and McCracken, 2010). The literature of conditional and unconditional testing procedures generally explores either the necessary assumptions to conduct the test or the asymptotic result of the different tests. Both DM and GW tests are well-accepted test for comparison of forecasting models. Nonetheless, it is very difficult to select one test as optimal across different datasets. This is the main reason for comparing all models by these two leading tests in the literature.

6. Trading Performance

Generating profitable trading signals is the main focal point of every trader. Trading profitability is not necessarily aligned with statistical accuracy. In this application, the trading performance evaluation of all models is done through a simple trading strategy. Namely, we go 'long' and 'short' when the forecasted return is positive and negative respectively. A 'long' or 'short' position means that we buy or sell respectively the ETF under study at the current price. Transaction costs severely impede the success of daily trading strategies, but ETFs offer investors the opportunity to trade stock indices with low transaction costs. In our case, the expense ratios for the all ETFs do not exceed 0.5% per annum.[10] The out-of-sample performance of all models is presented in the following table. The trading performance measures are given in appendix C.

[10] See, www.ishares.com/us/index

Table 8: Out-of-sample trading performance of every model for each ETF

Note: The table reports the annualized returns after transaction costs of every model and its respective information ratio in the parenthesis. Best1 and Best2 refer to the best individual predictors in terms of statistical and trading performance respectively in the in-sample period (as outlined in table 5).

	Market	ETF	RW	$Best_1$	$Best_2$	*v*SVR	GA-*v*SVR	KH-LSVR	RKH-LSVR
F1	US Currency	UUP	-1.88% (-0.95)	1.84% (0.74)	3.38% (1.44)	5.20% (1.77)	6.33% (1.82)	7.70% (2.05)	8.42% (2.32)
	US Stock	DIA	-1.37% (-0.45)	4.40% (1.38)	5.64% (1.71)	7.05% (1.84)	7.68% (2.14)	8.41% (2.32)	9.35% (2.64)
	EU Currency	FXB	-3.02% (-0.95)	1.25% (0.48)	2.37% (1.32)	4.28% (1.45)	5.15% (1.60)	6.90% (1.82)	7.25% (1.94)
	EU Stock	VGK	-2.88% (-0.76)	3.09% (1.27)	4.80% (1.62)	5.97% (1.80)	6.48% (1.92)	8.26% (2.17)	8.77% (2.51)
	Commodity: Oil	USO	-1.22% (-0.34)	5.56% (1.65)	7.25% (1.78)	8.13% (1.91)	8.71% (2.23)	9.35% (2.48)	10.22% (2.79)
	Commodity: Gold	IAU	-2.11% (-0.58)	6.33% (1.76)	7.48% (1.89)	8.69% (2.10)	9.15% (2.48)	9.62% (2.70)	10.82% (2.92)
F2	US Currency	UUP	-2.57% (-0.75)	1.35% (0.42)	2.66% (1.15)	3.51% (1.28)	4.23% (1.54)	5.84% (1.62)	7.09% (1.82)
	US Stock	DIA	-1.85% (-0.56)	3.35% (1.26)	5.48% (1.59)	6.90% (1.75)	7.35% (1.80)	8.50% (1.97)	9.15% (2.29)
	EU Currency	FXB	-3.28% (-0.94)	0.65% (0.12)	1.65% (0.85)	2.05% (1.12)	3.20% (1.22)	4.85% (1.38)	6.38% (1.64)
	EU Stock	VGK	-2.09% (-0.82)	2.42% (0.63)	4.51% (1.40)	5.68% (1.57)	6.35% (1.74)	7.62% (1.83)	8.20% (2.15)
	Commodity: Oil	USO	-1.54% (-0.48)	4.23% (1.55)	6.33% (1.63)	7.28% (1.89)	7.92% (1.95)	9.28% (2.21)	9.85% (2.37)
	Commodity: Gold	IAU	-1.32% (-0.47)	6.12% (1.70)	7.32% (1.84)	8.35% (1.94)	9.08% (2.34)	9.47% (2.53)	10.40% (2.82)
F3	US Currency	UUP	-1.24% (-0.62)	3.35% (1.55)	4.57% (1.70)	6.15% (1.81)	6.92% (1.94)	8.12% (2.22)	8.84% (2.47)
	US Stock	DIA	-2.03% (-0.88)	6.02% (1.68)	7.10% (1.80)	7.66% (1.85)	8.12% (2.14)	9.02% (2.42)	9.60% (2.74)
	EU Currency	FXB	-1.67% (-0.70)	2.86% (1.45)	3.52% (1.55)	5.35% (1.69)	6.05% (1.75)	7.04% (1.87)	7.78% (1.99)
	EU Stock	VGK	-3.19% (-0.81)	4.23% (1.60)	5.35% (1.75)	6.47% (1.82)	7.61% (2.01)	8.33% (2.31)	9.24% (2.56)
	Commodity: Oil	USO	-2.74% (-0.85)	7.20% (1.75)	7.55% (1.82)	8.15% (1.95)	9.14% (2.36)	9.80% (2.55)	10.62% (2.86)
	Commodity: Gold	IAU	-0.95% (-0.30)	7.35% (1.80)	8.07% (1.92)	9.20% (2.14)	9.39 % (2.49)	10.01% (2.82)	10.95% (2.98)
F4	US Currency	UUP	-3.84% (-1.02)	4.35% (1.65)	5.38% (1.73)	6.48% (1.79)	7.14% (2.13)	7.99% (2.24)	9.20% (2.41)
	US Stock	DIA	-1.39% (-0.65)	6.22% (1.81)	7.16% (1.94)	7.69% (2.08)	8.30% (2.20)	9.15% (2.36)	9.95% (2.80)
	EU Currency	FXB	-2.52% (-0.89)	3.12% (1.60)	4.10% (1.65)	5.55% (1.73)	6.35% (1.80)	7.66% (1.94)	8.15% (2.19)
	EU Stock	VGK	-1.32% (-0.65)	5.46% 1.78)	6.45% (1.81)	6.79% (1.86)	7.56% (1.95)	8.08% (2.12)	9.42% (2.50)
	Commodity: Oil	USO	-1.56% (-0.61)	7.44% (1.82)	8.15% (2.02)	8.62% (2.27)	9.54% (2.42)	10.23% (2.64)	11.02% (2.92)
	Commodity: Gold	IAU	-3.44% (-1.43)	7.35% (1.92)	8.50% (2.19)	9.35% (2.35)	9.95% (2.63)	10.60% (2.92)	11.12% (3.20)

The above table shows that RKH-LSVR delivers the best trading performance for all series and periods under. The second best model in terms of annualized return and information ratios after transaction costs appears to be the KH-LSVR. For example, during F1 on average RKH-LSVR presents 0.77% higher annualized returns and 0.26 higher information ratio after transaction costs. Both KH-LSVR hybrids consistently outperform their genetic counterpart in terms of profitability. In the same period, the KH methods outperform GA-*v*SVR on average by 1.51% and 0.36 in terms of profits and information ratios. These results suggest that our trading strategy benefits from the application of the KH and RKH optimization to the locally weighted SVR process. Finally, the *v*SVR provides higher profits and

information ratios than all individual predictors (linear or non-linear). Accounting for both the performance of $Best_1$ and $Best_2$ in period F1, individual predictors are found inferior to *v*SVR on average by 2.10% annualized returns and 0.39 information ratio. The above results are similar also in the rest of the forecasting exercises. Table 9 below examines the average trading performances of all models for all forecasting exercises.

Table 9: Average trading performances per forecasting exercise

Note: The table reports the average annualized returns after transaction costs of all models. Their information ratios are presented in the parenthesis. The Total Average corresponds to the average trading results over all ETFs under study. US and EU Averages refer to the average trading performance of all models over UUP, DIA and FXB, VGK respectively. Currency and Stock Averages refer to the average trading performance of all models over UUP, FXB and DIA, VGK respectively. Commodities Average refers to the average trading performance of all models over US and IAU.

	F1	F2	F3	F4
Total Average	5.45% (1.54)	4.81% (1.38)	6.07% (1.67)	6.32% (1.72)
US Average	4.81% (1.38)	4.36% (1.23)	5.87% (1.63)	5.98% (1.68)
EU Average	4.19% (1.30)	3.44% (0.99)	4.93% (1.49)	5.35% (1.53)
Currency Average	3.94% (1.20)	2.69% (0.89)	4.83% (1.48)	4.94% (1.50)
Stock Average	5.40% (1.58)	5.11% (1.33)	5.97% (1.64)	6.39% (1.71)
Commodities Average	7.01% (1.84)	6.63% (1.70)	7.41% (1.88)	7.63% (1.95)

As it turns out, the total average trading performance is worse in periods 2010-2012 and especially F2. The best results are provided during F4. This is expected since F2 includes the peak of the global financial crisis, while the effects of the crisis are minimized during the end of 2015 (F4). ETFs tracking the performance of the US markets are found to be more profitable than EU ones throughout both periods. This is also not surprising because the US economy was affected less by the European sovereign debt crisis. Additionally, the US ETFs recover more during 2013-2015. Examining the results in terms of different markets, currency ETFs present substantially lower annualized returns during 2010-2012. The pattern is similar but less strong when it comes to stock ETFs, while commodity ones appear less affected by the crisis. Nonetheless, they also seem to perform better in terms of annualized returns and information ratios during the aftermath of the European crisis.

The outcomes of the above trading exercises show that all models generate profits even in volatile economic periods, while complex techniques prove to be more successful than the simple ones. It is also worth noting that the trading performance of all models seems to vary through time. These two observations are consistent with one of the hypothesis of the AMH. Namely, the hypothesis that states that the performance of trading models varies through time and it deteriorates in times of market turbulence.

The next set of results refer to the decomposition of the trading performance of our models in the out-of-sample periods. This will allow us to test one further main implication of the AMH. The hypothesis that the profitability of all models is diminishing through time (else trading models are not robust in the

long run). Additionally, it will be interesting to see the "decay rate" of the profitability of the competing models in the different periods. AMH implies that the rate should be higher when the market is in crisis. Table 11 presents the monthly trading performance of all models and periods for the most profitable ETFs tracking currencies, stock indices and commodities. Namely, we will analyse UUP, DIA and IAU[11]

Table 10: Monthly Out-of-sample Trading Performance of selected ETFs

Note: The table reports the monthly annualized returns after transaction costs of every model. UUP, DIA and IAU cases are presented as they are the most profitable currency, stock and commodity ETFs respectively (on average).

		F1 (Out-of-sample)						F2 (Out-of-sample)					
ETF	**Models**	**Jan**	**Feb**	**Mar**	**Apr**	**May**	**Jun**	**Jul**	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**
UUP	RW	-3.44%	-0.98%	-3.12%	-2.10%	-0.22%	-1.42%	-3.17%	-2.23%	0.52%	-6.57%	1.22%	-5.21%
	$Best_1$	4.12%	2.98%	3.15%	1.15%	0.71%	-1.05%	3.45%	2.65%	2.52%	0.92%	-0.08%	-1.35%
	$Best_2$	7.38%	4.58%	2.99%	2.35%	1.68%	1.28%	6.25%	4.68%	2.60%	1.53%	0.94%	-0.04%
	*v*SVR	8.59%	7.22%	5.62%	4.26%	3.30%	2.19%	7.54%	4.94%	3.95%	2.27%	1.25%	1.12%
	GA-*v*SVR	9.47%	9.21%	5.71%	5.63%	4.12%	3.82%	10.55%	8.45%	2.35%	1.95%	1.92%	0.13%
	KH-LSVR	10.89%	10.24%	8.63%	6.74%	5.84%	3.85%	11.34%	9.56%	4.56%	3.76%	3.45%	2.39%
	RKH-LSVR	12.25%	12.48%	8.54%	7.20%	5.92%	4.12%	11.93%	9.10%	8.19%	6.48%	5.20%	1.65%
DIA	RW	-1.72%	-1.23%	0.36%	-2.34%	-0.88%	-2.43%	-2.24%	-1.27%	0.86%	-0.97%	-2.24%	-5.25%
	$Best_1$	6.56%	5.57%	4.47%	4.18%	3.30%	2.29%	5.52%	5.25%	3.89%	3.81%	1.29%	0.32%
	$Best_2$	7.35%	6.67%	6.14%	5.54%	4.87%	3.25%	7.98%	6.23%	5.36%	4.87%	4.95%	3.46%
	*v*SVR	10.25%	9.23%	8.14%	6.55%	5.05%	3.10%	9.12%	8.45%	8.12%	6.58%	5.28%	3.82%
	GA-*v*SVR	12.23%	10.94%	8.84%	6.55%	4.26%	3.25%	11.30%	10.84%	8.42%	5.67%	4.43%	3.45%
	KH-LSVR	12.84%	10.48%	9.87%	8.24%	4.48%	4.56%	12.45%	11.87%	9.48%	8.03%	6.47%	4.12%
	RKH-LSVR	13.02%	12.64%	10.87%	9.14%	5.59%	4.84%	12.66%	11.95%	11.44%	9.13%	6.35%	3.38%
IAU	RW	-2.35%	-1.05%	-2.20%	-1.80%	-1.15%	-4.12%	-2.10%	0.95%	-1.75%	0.05%	-3.70%	-1.34%
	$Best_1$	8.81%	8.15%	7.02%	5.89%	4.99%	3.14%	7.89%	7.05%	6.54%	5.99%	5.23%	4.02%
	$Best_2$	8.84%	8.42%	7.94%	7.60%	6.19%	5.86%	9.40%	9.13%	8.45%	7.62%	5.21%	4.12%
	*v*SVR	10.52%	9.53%	9.12%	8.84%	7.42%	6.70%	11.25%	10.85%	9.25%	8.17%	5.74%	4.85%
	GA-*v*SVR	10.45%	10.95%	9.68%	8.66%	8.26%	6.92%	11.56%	10.85%	9.84%	9.42%	7.70%	5.12%
	KH-LSVR	11.85%	11.02%	9.84%	9.23%	8.17%	7.62%	12.05%	11.42%	10.24%	9.25%	7.92%	5.95%
	RKH-LSVR	12.25%	11.74%	11.41%	10.23%	9.84%	9.46%	13.02%	11.25%	10.36%	9.45%	9.32%	9.02%
		F3 (Out-of-sample)						**F4 (Out-of-sample)**					
ETF	**Models**	**Jan**	**Feb**	**Mar**	**Apr**	**May**	**Jun**	**Jul**	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**
UUP	RW	-3.47%	-1.85%	-1.54%	1.15%	-1.38%	-0.34%	-5.67%	-5.48%	-0.22%	-2.38%	-3.92%	-5.38%
	$Best_1$	6.25%	5.21%	3.23%	2.56%	1.52%	1.34%	6.84%	5.45%	4.49%	3.26%	3.28%	2.78%
	$Best_2$	8.06%	6.32%	5.08%	3.53%	2.99%	1.42%	8.17%	6.41%	5.23%	5.28%	4.06%	3.15%
	*v*SVR	9.24%	7.58%	6.47%	5.87%	4.41%	3.30%	9.05%	8.12%	7.40%	6.24%	4.98%	3.07%
	GA-*v*SVR	10.54%	9.58%	6.58%	6.05%	5.72%	3.05%	9.98%	8.87%	8.49%	6.58%	6.10%	2.84%
	KH-LSVR	10.97%	9.97%	8.21%	7.23%	6.63%	5.68%	11.25%	10.94%	9.41%	7.52%	5.78%	3.06%
	RKH-LSVR	11.34%	10.58%	9.48%	7.89%	7.64%	6.13%	11.82%	11.25%	10.84%	9.63%	8.41%	3.25%
DIA	RW	0.85%	-4.86%	-1.96%	-3.05%	-2.34%	-0.84%	3.84%	-2.57%	-1.23%	0.42%	-6.45%	-2.35%
	$Best_1$	8.84%	8.45%	7.85%	5.10%	3.48%	2.37%	9.26%	9.28%	8.42%	6.45%	3.52%	0.38%
	$Best_2$	9.94%	8.56%	8.12%	6.12%	5.58%	4.28%	9.66%	8.85%	8.74%	8.25%	4.32%	3.15%
	*v*SVR	10.23%	9.04%	9.98%	7.25%	6.33%	4.85%	10.45%	9.52%	9.05%	8.14%	5.68%	3.30%
	GA-*v*SVR	11.95%	10.85%	10.32%	7.56%	4.69%	3.34%	11.23%	10.23%	9.56%	7.95%	6.58%	4.25%
	KH-LSVR	12.05%	11.24%	9.21%	8.23%	7.23%	6.15%	11.85%	11.02%	10.18%	9.26%	7.42%	5.15%
	RKH-LSVR	12.95%	12.25%	10.28%	8.89%	7.75%	5.45%	12.25%	12.35%	10.42%	9.48%	8.45%	6.74%
IAU	RW	0.55%	-1.89%	0.14%	-2.45%	-1.85%	-0.17%	-2.65%	-6.26%	1.32%	-2.64%	-6.08%	-4.31%
	$Best_1$	9.85%	9.12%	8.25%	7.23%	5.49%	4.14%	8.86%	8.45%	7.95%	7.26%	6.25%	5.35%
	$Best_2$	10.12%	9.15%	8.64%	8.40%	6.30%	5.80%	10.23%	9.85%	9.47%	8.53%	7.03%	5.90%
	*v*SVR	10.62%	10.15%	9.25%	9.10%	8.58%	7.48%	10.60%	10.35%	10.13%	9.45%	8.74%	6.85%
	GA-*v*SVR	11.20%	10.85%	10.45%	9.84%	7.89%	6.10%	10.96%	10.52%	10.27%	9.74%	9.34%	8.84%
	KH-LSVR	11.75%	11.09%	10.74%	9.94%	9.05%	7.48%	12.30%	12.07%	11.57%	9.98%	9.12%	8.53%
	RKH-LSVR	12.56%	11.84%	10.78%	10.46%	10.23%	9.84%	12.56%	12.94%	12.24%	10.84%	9.14%	9.01%

[11] The results of the remaining three ETFs are not presented here for the sake of space. The pattern of their returns is similar with those presented in table 11.

The above results show that the profitability of all models is declining as time passes by. The declining speed seems to be increased after the third month of the out-of-sample. In order to further examine this behavior, a decay speed is calculated as below:

$$\text{decay rate} = \frac{\text{current month average return - previous month average return}}{\text{previous month average return}} \tag{29}$$

This decay rate and its average for each forecasting exercise is presented in Table 11 below.

Table 11: Monthly Returns Decay Speed

*Note: Columns (1)-(5) present the decay rate for the second to the sixth month of the out-of-sample. The last column is the average, while in the parenthesis is the p-value of the test of equal means between the current exercise and the previous one for each ETF. RW is excluded from these calculations. ***, ** and * denote a rejection of the null hypothesis at the 1%, 5% and 10% significance level respectively.*

ETF	Forecasting Exercise	(1)	(2)	(3)	(4)	(5)	Average
UUP	F1	-11.37%	-12.84%	-18.10%	-21.08%	-34.12%	-15.85%
	F2	-22.88%	-38.62%	-39.04%	-45.01%	-79.24%	-36.39% (0.005***)
	F3	-13.67%	-20.55%	-19.56%	-9.62%	-33.16%	-15.85% (0.009***)
	F4	-11.88%	-13.65%	-9.88%	-15.49%	-55.20%	-12.73% (0.377)
DIA	F1	-7.80%	-8.97%	-13.82%	-21.47%	-22.72%	-14.95%
	F2	-11.52%	-18.43%	-18.45%	-44.47%	-65.52%	-31.68% (0.043**)
	F3	-8.44%	-7.67%	-20.61%	-18.75%	-21.59%	-15.41% (0.062*)
	F4	-5.33%	-7.97%	-12.13%	-27.38%	-36.14%	-17.79% (0.297)
IAU	F1	-6.64%	-8.03%	-8.29%	-11.06%	-11.52%	-9.11%
	F2	-5.09%	-9.69%	-8.74%	-15.60%	-14.55%	-10.73% (0.097*)
	F3	-5.90%	-7.58%	-6.40%	-15.52%	-14.09%	-9.90% (0.119)
	F4	-3.03%	-3.97%	-9.46%	-13.08%	-11.36%	-8.18% (0.114)

Table 12 shows that decay rate is generally increasing through time. It is worth noting that on average the decay rate is higher on F2 throughout the four exercises. The returns appear to erode quicker in the case of the currency ETF, while for the commodity ETF the returns decrease with the least speed. The p-values suggest that the decay rates for UUP and DIA are statistical different between F1, F2 and F2, F3. It seems that the decay rates during the European debt crisis is statistically different and higher. Although the results for the case of IAU follow a similar pattern as with UUP and DIA, the differences are statistically insignificant. This is expected since commodity markets where the least affected market during the crisis period. These results allow us to conclude that for the periods under study our models are less robust, when the underlying forecasted market is in crisis. Under normal or near to normal market conditions the decay rates seem similar.

7. Conclusions

In this research a hybrid RKH-LSVR model is introduced. The RKH algorithm is a novel metaheuristic optimization technique inspired by the behaviour of krill herds. The RKH is used to optimize the LSVR parameters by balancing the search between local and global optima. The proposed model is evaluated through three different fitness functions, while its statistical and trading performance is benchmarked against a set of traditional SVR structures, non-linear and linear models and a RW. The inputs of the SVR models are selected through a large pool of linear and non-linear predictors and PCA analysis. All models are applied in four forecasting and trading exercises over six ETFs during the period 2010-2015. The purpose of the trading applications is to test the robustness of the models under study and to provide empirical evidence in favour of the AMH.

In terms of the results, RKH-LSVR architectures outperform their counterparts in terms of statistical accuracy and trading efficiency. The trading application provides evidences in favour of the AMH. This work should go forward on convincing researchers, practitioners and academics to explore further hybrid SVR techniques. It should also serve as caution on the implications of the AMH and the robustness of trading models.

References

Aguilar-Rivera, R., Valenzuela-Rendón, M. & Rodríguez-Ortiz, J.J. (2015) Genetic algorithms and Darwinian approaches in financial applications: A survey, Expert Systems with Applications, 42 (21), pp. 7684-7697.

Basak, D., Pal, S., & Patranabis, D. C. (2007) Support vector regression. Neural Information Processing-Letters and Reviews, 11 (10), pp. 203-224.

Bernstein, F., Li, Y. & Shang, K. (2015) A simple heuristic for joint inventory and pricing models with lead time and backorders. Management Science. Forthcoming.

Bolaji, A. L. A., Al-Betar, M. A., Awadallah, M. A., Khader, A. T., & Abualigah, L. M. (2016) A comprehensive review: Krill Herd algorithm (KH) and its applications. Applied Soft Computing, 49, pp. 437-446.

Chan, K.S.& Tong, H. (1986) On estimating thresholds in autoregressive models, Journal of Time Series Analysis, 7 (3), pp. 178–190.

Chang, T.J., Meade, N., Beasley, J.E. & Sharaiha, Y.M. (2000) Heuristics for cardinality constrained portfolio optimization, Computers and Operations Research, 27 (13), pp. 1271–1302.

Cherkassky,V. & Ma,Y. (2004) Practical selection of SVM parameters and noise estimation for SVM regression, Neural Networks, 17 (1), pp. 113-126.

Clark, T.E. & McCracken, M.W. (2010) Averaging forecasts from VARs with uncertain instabilities. Journal of Applied Econometrics, 25(1), pp. 5-29.

Cleveland, W.S. & Devlin, S.J. (1988) Locally Weighted Regression: an approach to regression analysis by ocal fitting, Journal of the American Statistical Association, 83 (403), pp. 596-610.

Diebold, F.X. & Mariano, R.S. (1995) Comparing predictive ability, Journal of Business & Economic Statistics, 13 (3), pp. 253-263

Dolvin, D. (2010) S&P ETFs: arbitrage opportunities and market forecasting, The Journal of Index Investing, 1(1), pp. 107–116.

Dunis, C.L., Laws, J. & Sermpinis, G. (2011) Higher order and recurrent neural architectures for trading the EUR/USD exchange rate, Quantitative Finance, 11 (4), pp. 615-629.

Dunis, C.L. & Nathani, A. (2007) Quantitative trading of gold and silver using non-linear models, Neural Network World, 16 (2), pp. 93-111.

Elman, J.L. (1990) Finding structure in time, Cognitive Science, 14 (2), pp. 179-211.

Ferreira, C. (2001) Gene Expression Programming: A New Adaptive Algorithm for Solving Problems. Complex Systems, 13(2), pp. 87-129.

Ferri, R.A. (2009) The ETF Book: All you need to know about Exchange-Traded Funds, John Wiley & Sons: Hoboken, New Jersey.

Fix, E. & Hodges, J.L. (1951) Discriminatory analysis, nonparametric discrimination: Consistency properties. Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas.
Gandomi, A.H. & Alavi, A.H. (2012) Krill herd: A new bio-inspired optimization algorithm, Communications in Nonlinear Science and Numerical Simulation, 17 (12), pp. 4831-4845.
Giacomini, R. & White, H. (2006) Tests of conditional predictive ability, Econometrica, 74(6), pp. 1545–1578.
Gilli, M., Maringer, D & Winker, P. (2008) Applications of heuristics in finance. In: Handbook On Information Technology in Finance, Seese, D., Weinhardt,C. & Schlottman, F. (Eds.), International Handbooks on Information Systems. Springer, Germany, pp. 635–653.
Goldberg, D.E. (1989) Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley.
Ghosh, J. & Shin, Y. (1991) The Pi-Sigma Network: An efficient higher-order neural networks for pattern classification and function approximation, Proceedings of International Joint Conference of Neural Networks, 1, pp. 13-18.
Hall, N.G. & Posner, M.E. (2007) Performance prediction and preselection for optimization and heuristic solution procedures. Operations research,55(4), pp.703-716.
Holland, J. (1975) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control and artificial intelligence, Cambridge, Mass: MIT Press.
Hsu, S.H., Hsieh, J.J.P.A., Chih, T.C. & Hsu, K.C. (2009) A two-stage architecture for stock price forecasting by integrating self-organizing map and support vector regression, Expert Systems with Applications, 36 (4), pp. 7947-7951.
Huang, K., Yang, H., King, I. & Lyu, M. R. (2006) Local support vector regression for financial time series prediction, Proceedings of International Joint Conference on Neural Networks, pp. 1622-1627.
Huang, S.C., Chuang, P.J., Wu, C.F. & Lai, H.J. (2010) Chaos-based support vector regressions for exchange rate forecasting, Expert Systems with Applications, 37(12),pp. 8590-8598.
Huang, K., Zhou, Y., Wu, X., & Luo, Q. (2016) A Cuckoo Search Algorithm With Elite Opposition-Based Strategy. Journal of Intelligent Systems, 25(4), pp. 567-593.
Jarque, C.M. & Bera, A.K. (1980) Efficient tests for normality, homoscedasticity and serial independence of regression residuals, Economics Letters, 6 (3), pp. 255-259.
Jiang, H. & He, W. (2012) Grey relational grade in local support vector regression for financial time series prediction, Expert Systems with Applications, 39 (3), pp. 2256-2262.
Jolliffe, I. (2002) Principal component analysis. John Wiley & Sons, Ltd.
Kao,L.J.,Chiu,C.C.,Lu,C.J., Yang, J.L. (2013) Integration of nonlinear independent component analysis and support vector regression for stock price forecasting, Neurocomputing, 99(1), pp. 534-542.
Karaboga, D. & Basturk, B. (2008) On the performance of artificial bee colony (ABC) algorithm, Applied Soft Computing, 8 (1), pp. 687-697.
Koza, J., & Poli, R., (2005) Genetic Programming, in: Burke, E.K., Kendall, G. (eds.), Search Methodologies, Introductory Tutorials in Optimization and Decision Support Techniques, Springer, pp.127-164.
Lee, DE., Song, JH., Song, SO., Yoon, ES. (2005), Weighted support vector machine for quality estimation in the polymerization process, Industrial & Engineering Chemistry Research, 44(7), pp. 2101-2105.
Li, X., Zhang, J., Yin, M. (2014) Animal migration optimization: an optimization algorithm inspired by animal migration behavior, Neural Computing and Applications, 24 (7-8), pp. 1867-1877.
Liang, J.J., Qin, A.K., Suganthan, P.N. & Baskar, S. (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Transactions on Evolutionary Computations, 10 (3), pp. 281–295.
Lin, K.P. & Pai, P.F. (2010) A fuzzy support vector regression model for business cycle predictions, Expert Systems with Applications, 37 (7), pp. 5430-5435.
Lin, C.J. & Teräsvirta, T. (1994) Testing the constancy of regression parameters against continuous structural changes, Journal of Econometrics, 62 (2), pp. 211–228.
Lo, A. W. (2000) Finance: A selective survey. Journal of the American Statistical Association, 95 (450), 629-635.
Lo, A.W. (2004) The adaptive markets hypothesis, Journal of Portfolio Management, 30, pp. 15–29.
Lu, C.J., Lee, T.S., Chiu, C.C. (2009) Financial time series forecasting using independent component analysis and support vector regression, Decision Support Systems, 47 (2), pp. 115-125.
Martí, R., Gallego, M., Abraham Duarte, M. & Pardo, E.G. (2013) Heuristics and metaheuristics for the maximum diversity problem, Journal of Heuristics, 19 (4), pp. 591-615.
Marshall, B.R., Nguyen, N.H & Visaltanachoti, N. (2013) ETF arbitrage: Intraday evidence, Journal of Banking & Finance, 37 (9), pp. 3486-3498.
Oreski, S. & Oreski, G. (2014) Genetic algorithm-based heuristic for feature selection in credit risk assessment, Expert Systems with Applications, 41 (4), pp. 2052-2064.
Pai, P.F., Lin, C.S., Hong, W.C. & Chen, C.T. (2006) A hybrid support vector machine regression for exchange rate prediction, International Journal of Information and Management Sciences, 17 (2), pp. 19-32.
Park, C.H. and Irwin, S.H. (2007) What do we know about profitability of technical analysis?, Journal of Economic Surveys, 21 (4), pp. 786–826.
Pesaran, M.H. & Timmerman, A.G. (1992) A simple nonparametric test of predictive performance, Journal of Business and Economic Statistics, 10(4), pp. 461-465.
Popescu, D. & Crama, P. (2015) Advertising revenue optimization in live television broadcasting. Management Science, Forthcoming.
Ruiz, A. & Lopez-de-Teruel, P.E. (2001) Nonlinear kernel-based statistical pattern analysis, IEEE Transactions on Neural Networks, 12 (1), pp. 16-32.
Schölkopf, B., Bartlett, P., Smola, A., Williamson, R. (1999) Shrinking the tube: a new support vector regression algorithm, In: Kearns, M.J., (ed.), Advances in neural information processing systems 11. Cambridge, Mass, MIT Press, pp. 330-336.

Shapiro, A.F. (2000) A Hitchhiker's guide to the techniques of adaptive non-linear models. Insurance: Mathematics and Economics, 26 (2-3), pp. 119-132.
Sermpinis, G., Theofilatos, K.A, Karathanasopoulos, A.S., Georgopoulos, E.F. & Dunis, C.L. (2013) Forecasting foreign exchange rates with adaptive neural networks using radial-basis functions and Particle Swarm Optimization, European Journal of Operational Research, 225 (3), pp. 528–540.
Sermpinis, G., Stasinakis, C., Theofilatos, K. & Karathansopoulos, A. (2015) Modeling, forecasting and trading the EUR exchange rates with hybrid rolling genetic algorithms – support vector regression forecast combinations, Journal of Operational Research, 247(3), pp. 831-846.
Suykens, J.A.K., De Brabanter, J., Lukas, L. & Vandewalle, J. (2002) Weighted least squares support vector machines: robustness and sparse approximation, Neurocomputing, 48 (1-4), pp. 85-105.
Talbi, E.G. (2009) Metaheuristics: from design to implementation. Hoboken, New Jersey, USA: John Wiley & Sons.
Tenti, P. (1996) Forecasting foreign exchange rates using recurrent neural networks, Applied Artificial Intelligence, 10 (6), pp. 567-582.
Timmermann, A. (2006) Forecast Combinations, In: Handbook of Economic Forecasting, Elliott, G. Granger, C.W.J. &. Timmermann, A. (Eds), 1, pp. 135-196.
Trafalis, T.B., Ince, H. (2000) Support vector machine for regression and applications to financial forecasting, IJCNN, Neural Networks, IEEE, 6, pp. 348-353.
Turmon, M. J. (1998) Machine Learning and Statistics: The Interface. Journal of the American Statistical Association, 93 (442), pp. 833-835.
Vapnik, V. (1995) The nature of statistic learning theory, Springer-Verlag, New York.
Wagner, D. (2011) Trading ETFs: Gaining an edge with technical analysis, John Wiley & Sons: Hoboken, New Jersey.
Wang,G.G., Guo,L., Gandomi,A.H., Hao,G.S. & Wang,H. (2014) Chaotic Krill Herd algorithm, Information Sciences, 274 (1), pp. 17-34.
Wu, S. & Akbarov, A. (2011) Support vector regression for warranty claim forecasting, European Journal of Operational Research, 213 (1), pp. 196-204.
Wu, CH., Tzeng, GH., Goo, JY. & Fang, WC. (2007) A real-valued genetic algorithm to optimize the parameters of support vector machine for predicting bankruptcy, Expert Systems with Applications, 32 (2), pp. 397-408.
Wu, Y. & Liu, Y. (2007) Robust truncated hinge loss support vector machines, Journal of the American Statistical Association, 102(479), pp. 974-983.
Xu, Q., Wang, L., Wang, N., Hei, X., & Zhao, L. (2014) A review of opposition-based learning from 2005 to 2012. Engineering Applications of Artificial Intelligence, 29, pp. 1-12.
Yang, H., Huang, K., King, I. & Lyu, M.R. (2009) Localized support vector regression for time series prediction, Neurocomputing, 72 (10–12), pp. 2659-2669.
Yang, X.S. (2010) Firefly algorithm nature-inspired metaheuristic algorithms. United Kingdom: Luniver Press.
Yang, X.S. & Gandomi, A.H. (2012) Bat algorithm: a novel approach for global engineering optimization, Engineering Computations, 29 (5), pp. 464–483.
Yeh, C.Y., Huang, C.W. & Lee, S.J. (2011) A multiple-kernel support vector regression approach for stock market price forecasting, Expert Systems with Applications, (3), pp.2177-2186.
Yuan, FC. (2012) Parameters optimization using genetic algorithms in support vector regression for sales volume forecasting, Applied Mathematics, 3 (1), pp. 1480-1486.
Zhang, G., Hu, M.Y., Patuwo, B.E. & Indro, D.C. (1999) Artificial neural networks in bankruptcy prediction: General framework and cross-validation analysis, European Journal of Operational Research, 116 (1), pp. 16-32.
Zhang, L., Zhou, W. & Jiao, L. (2004) Wavelet support vector machine, IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 34 (1), 34-39.

Appendix

A. Predictors' pool

This appendix section is a short description of the linear and non-linear models used to populate the individual forecast pools.

A.1 Linear Predictors

The linear models used are SMA, EMA, AR, ARMA, ROC and PPI. Their specifications are provided in the following table. In total, the linear models' forecasts sum up to 300.

Table A.1: The specification of the linear models

LINEAR MODELS	DESCRIPTION	TOTAL INDIVIDUAL FORECASTS
SMA (q)	$E(R_t) = (R_{t-1} + ... + R_{t-q}) / q$ *Where:* • q=3...25	23
EMA (q')	$E(R_t) = \frac{R_{t-1} + (1-a')R_{t-2} + ... + (1-a')^{q'-1} R_{t-q'}}{a' + (1-a') + ... + (1-a')^{q'-1}}$ *Where:* • q'=3...25 • a'=2/(1+N_{days}), N_{days} is the number trading days	23
AR (q'')	$E(R_t) = \beta_0 + \sum_{i''=1}^{q''} \beta_{i''} R_{t-i''}$ *Where:* • q''=1,...,20 • $\beta_0, \beta_{i'}$ the regression coefficients	20
ARMA (m', n')	$E(R_t) = \bar{\varphi}_0 + \sum_{j'=1}^{m'} \bar{\varphi}_{j'} R_{t-j'} + \bar{a}_0 + \sum_{k'=1}^{n'} \bar{w}_{k'} \bar{a}_{t-k'}$ *Where:* • m', n'=1,..,15 • $\bar{\varphi}_0, \bar{\varphi}_{j'}$ the regression coefficients • $\bar{a}_0, \bar{a}_{t-k'}$ the residual terms • $\bar{w}_{k'}$ the weights of the residual terms	210
ROC (p')	$E(R_t) = 100[1-(R_{t-1} / R_{t-p'})]$ *Where:* • p'=3,..,25	23
PPI	$\left\{ \begin{matrix} PivotP_{t-1} = (H_{t-1} + L_{t-1} + CP_{t-1}) / 3 \\ E(R_t) = (PivotP_{t-1} - CP_{t-1}) / CP_{t-1} \end{matrix} \right\}$ *Where:* • $PivotP_{t-1}$ the pivot point for t-1 • $H_{t-1}, L_{t-1}, CP_{t-1}$ the high, low and closing price for t-1	1

A.2 Non-linear Predictors

A.2.1. Smooth Transition Autoregressive Models (STAR)

STAR as proposed by Chan and Tong (1986) are extensions of the ARs. The STAR combines two AR models with a function that defines the degree of non-linearity (smooth transition function).

$$E(R_t) = \Phi_1' X_t (1 - F'(z_t', \zeta', \lambda')) + \Phi_2' X_t F'(z_t', \zeta', \lambda') + u_t' \quad (A.1)$$

Where: $\Phi_i' = (\tilde{\varphi}_{i,0}, \tilde{\varphi}_{i,1}, ...\tilde{\varphi}_{i,p}), i = 1,2$ and $\tilde{\varphi}_{i,0}, \tilde{\varphi}_{i,1}, ...\tilde{\varphi}_{i,p}$ the regression coefficients of the two AR models, $X_t = (1, \tilde{\chi}_t')'$ with $\tilde{\chi}_t' = (R_{t-1}, ..., R_{t-p''}), 0 \le F'(z'_t, \zeta', \lambda') \le 1$ the smooth transition function, $z'_t = R_{t-d'}, d' > 0$ the lagged endogenous transition variable, ζ' the parameter that defines the smoothness of the transition between the two regimes, λ' the threshold parameter and $u_{t'}$ the error term. In this case, we estimate two-regime logistic (LSTAR) and exponential (ESTAR) STARs (Lin and Teräsvirta, 1994). For both models the orders 1 to 20 are explored.

A.2.2. Nearest Neighbors Algorithm (*k*-NN)

Nearest Neighbors (*k*-NN) is a non-linear and non-parametric forecasting method based on the work of Fix and Hodges (1951). It is based on the idea that pieces of time series in the past have patterns which might have resemblance to pieces in the future. Similar patterns of behavior are located in terms of nearest neighbors using a distance called the Euclidean distance and these patterns are used to predict behavior in the immediate future. We follow the guidelines of Dunis and Nathani (2007) to define the parameters. The optimal set of parameters is selected based on the highest trading performance in the in-sample period.

A.2.3. Neural Networks (NNs)

In this analysis, five NN architectures are applied. The simpler and most popular is the MLP. A standard MLP has at least three layers. The first layer is called the input layer (the number of its nodes corresponds to the number of explanatory variables). The last layer is called the output layer (the number of its nodes corresponds to the number of response variables). An intermediary layer of nodes, the hidden layer, separates the input from the output layer. Its number of nodes defines the amount of complexity the model is capable of fitting. In addition, the input and hidden layer contain an extra node called the bias node. This node has a fixed value of one and has the same function as the intercept in traditional regression models. Normally, each node of one layer has connections to all the other nodes of the next layer.

The training of the network (which is the adjustment of its weights in a way that the network maps the input value of the training data to the corresponding output value) starts with randomly chosen weights and proceeds by applying a learning algorithm called back-propagation of errors (Shapiro, 2000). The maximum number of the allowed back-propagation iterations is optimized by maximizing a fitness function in the test dataset (see table 2) through a trial and error procedure. More specifically, the learning algorithm tries to find those weights which minimize an error function (normally the sum of all squared differences between target and actual values). Since networks with sufficient hidden nodes are able to learn the training data (as well as their outliers and their noise) by heart, it is crucial to stop the training procedure at the right time to prevent overfitting (this is called ‘early stopping’). This is achieved by dividing the dataset into 3 subsets respectively called the training and test sets used for simulating the

data currently available to fit and tune the model and the validation set used for simulating future values. The network parameters are then estimated by fitting the training data using the backpropagation of errors. The iteration length is optimized by maximizing the forecasting accuracy for the test dataset. Then the predictive value of the model is evaluated applying it to the validation dataset (out-of-sample dataset).

A Recurrent Neural Network is also applied. For an exact specification of recurrent networks, see Elman (1990). A simple recurrent network has an activation feedback which embodies short-term memory. In other words, the RNN architecture can provide more accurate outputs because the inputs are (potentially) taken from all previous values. Although RNN require substantially more computational time (see Tenti (1996), they can yield better results in comparison with simple MLPs due to the additional memory inputs. The third NN model included in the feature space is the Higher Order Neural Network (HONN). HONNs are able to simulate higher frequency, higher order non-linear data, and consequently provide superior simulations. For more information on HONNs see Dunis *et al.* (2011). Psi Sigma Networks (PSNs) are considered as a class of feed-forward fully connected HONNs. First introduced by Ghosh and Shin (1991), the PSN creation was motivated by the need to create a network combining the fast learning property of single layer networks with the powerful mapping capability of HONNs, while avoiding the combinatorial increase in the required number of weights. The order of the network in the context of PSN is represented by the number of hidden nodes. In a PSN the weights from the hidden to the output layer are fixed to 1 and only the weights from the input to the hidden layer are adjusted, something that greatly reduces the training time.

The last NN used is the ARBF-PSO. Its complexity, architecture and characteristics differ from the previous mentioned NNs. Compared to them, in the ARBF-PSO the parameters are optimized through a Particle Swarm Optimization (PSO)[12] algorithm. This protects the ARBF-PSO from the dangers of over-fitting and data snooping. However, the practitioner still needs to select the network's inputs as with the previous NNs. For a complete description of the ARBF-PSO see Sermpinis *et al.* (2013). Table B.2 summarizes the learning algorithm, hidden and output node activation functions for all previous structures.

Table A.2: Neural Network Design and Training Characteristics

Note: The input of every node is z_ψ, where $\psi = 1 \ldots n''$ and n'' is the number of nodes of the previous layer. The vector indicating the center of the Gaussian function is C' and σ' is the value indicating its width.

PARAMETERS	MLP	RNN	HONN	PSN	ARBF-PSO
Learning algorithm	Gradient descent	Gradient descent	Gradient descent	Gradient descent	Particle Swarm Optimization
Initialisation of weights	N(0,1)	N(0,1)	N(0,1)	N(0,1)	-
Hidden node activation function	$F(z_\psi) = 1/(1+e^{-z_\psi})$	$F(z_\psi) = 1/(1+e^{-z_\psi})$	$F(z_\psi) = 1/(1+e^{-z_\psi})$	$F(z_\psi) = \sum_{\psi=1}^{n''} z_\psi$	$\mathrm{F}(z_\psi) = \exp\left(\frac{\left\|z_\psi - C'\right\|^2}{2\sigma'^2}\right)$

[12] The PSO algorithm is a population based heuristic search algorithm based on the social behavior of birds within a flock (Liang et al, 2006).

Output node activation function	$F(z_\psi) = \sum_{\psi=1}^{n''} z_\psi$	$F(z_\psi) = \sum_{\psi=1}^{n''} z_\psi$	$F(z_\psi) = \sum_{\psi=1}^{n''} z_\psi$	$F(z_\psi) = 1/(1+e^{-z_\psi})$	$F(z_\psi) = \sum_{\psi=1}^{n''} z_\psi$

There is no formal theory behind the selection of the NN inputs and their characteristics, such as number of hidden neurons, learning rate, momentum and iterations. We conduct NN experiments and a sensitivity analysis on a pool of autoregressive and autoregressive-moving average terms of all series in the in-sample dataset. For example for the number of iterations, our experimentation started from 1.000 iterations and stopped at the 100.000 iterations, increasing in each experiment the number of iterations by 1.000. This is a very common approach in the literature (Tenti, 1996). Based on these experiments and the sensitivity analysis, the sets of variables selected are those that provide the higher trading performance for each network in the in-sample period.

A.2.4. Genetic Programming predictors

The final two non-linear models are the GP and GEP. These techniques are domain-independent problem solving methods based on the Darwinian principle of reproduction and survival of the fittest. GP, as class of GAs, creates an initial population of models and evolves it using genetic operators (crossover and mutation). The result is to perform mathematical expressions that best fit to the given input (data). When designing a GP algorithm the main focus is on optimizing execution time and limiting the 'bloat effect', a similar issue to over-fitting in NNs mentioned earlier. This genetic procedure creates superior offsprings, replacing the worst models (tournament losers), and rearranges the initial population for the next iteration. The iterations stop and the final forecast results are obtained when the model reaches the critical value of the termination criterion. GP holds a greater selection strength and genetic drift from a typical GA. The functionality aspects of GP and the genetic operators are described in detail by Koza and Poli (2005).

GEP is based on symbolic strings of fixed length that represent the genotype of an organism. Its chromosomes consist of multiple genes with equal lengths. Each gene includes a head (detailing symbols specific to functions and terminals) and a tail (only includes terminals). The set of terminals included within both the heads and tails of the chromosomes comprises constants and specific variables. Each gene holds the capacity to code for multiple and different expression trees. Valid expression trees are always generated when using GEP, while it can operate also when the first element of a gene is terminal. This is not guaranteed in GP. GEP is also able to code for sub-expression trees with interlinking functions in order to enable reproduction when multiple generations arise. In general, GEP is considered superior to GP because fitness is established through the genotype and phenotype of an individual based on its chromosomes and expression trees respectively. Ferreira (2001) provides details on the exact procedure of GEP.

B. Parameters and training characteristics

The table B.1 summarizes the training characteristics of the GA and both KH algorithms for all ETFs and forecasting exercises.

Table B.1: GA and KH training characteristics

	GA					KH / RKH				
	Forecasting Exercise	**1**	**2**	**3**	**4**	**Forecasting Exercise**	**1**	**2**	**3**	**4**
UUP	**Population Size**	50	50	50	50	**Population Size**	50	50	50	50
	Maximum Generations	900	900	900	900	Δt **,** Z_{cr}	22.32, 0.43	21.78, 0.66	19.40, 0.60	22.56, 0.44
DIA	**Population Size**	60	60	60	60	**Population Size**	60	60	60	60
	Maximum Generations	1000	1000	1000	1000	Δt **,** Z_{cr}	11.48, 0.64	23,56, 0.95	18.01, 0.52	16.02, 0.54
FXB	**Population Size**	55	55	55	55	**Population Size**	55	55	55	55
	Maximum Generations	1000	1000	1000	1000	Δt **,** Z_{cr}	10.19, 0.80	13.30, 0.78	21.14, 0.60	20.10, 0.31
VGK	**Population Size**	80	80	80	80	**Population Size**	80	80	80	80
	Maximum Generations	800	800	800	800	Δt **,** Z_{cr}	13.03, 0.65	15.84, 0.55	25.13, 0.39	18.14, 0.48
USO	**Population Size**	90	90	90	90	**Population Size**	90	90	90	90
	Maximum Generations	750	750	750	750	Δt **,** Z_{cr}	19.44, 0.76	18.05, 0.42	12.98, 0.80	20.98, 0.74
IAU	**Population Size**	60	60	60	60	**Population Size**	60	60	60	60
	Maximum Generations	850	850	850	850	Δt **,** Z_{cr}	24.40, 0.36	29.13, 0.66	26.90, 0.29	25.68, 0.50
ALL ETFs	**Selection Type**	Roulette Wheel Selection				**Foraging Speed**	0.02 ms^{-1}			
	Elitism	Best individual is kept in the next generation.				**Maximum Motion Speed**	0.01 ms^{-1}			
	Crossover Probability	0.9				**Maximum Diffusion Speed**	[0.002, 0.010] ms^{-1}			
	Mutation Probability	0.1				**Inertia Weights**	[0,1]			

Table B.2 presents the selected fitness functions for the proposed RKH-LSVR and its GA-*v*SVR and KH-LSVR benchmarks.

Table B.1: Selected Fitness Functions

	Series	**GA-*v*SVR**	**KH-LSVR**	**RKH-LSVR**		**Series**	**GA-*v*SVR**	**KH-LSVR**	**RKH-LSVR**
F1	SPY	$Fitness_1$	$Fitness_2$	$Fitness_3$	F2	SPY	$Fitness_1$	$Fitness_3$	$Fitness_3$
	QQQ	$Fitness_3$	$Fitness_3$	$Fitness_3$		QQQ	$Fitness_2$	$Fitness_3$	$Fitness_3$
	DIA	$Fitness_2$	$Fitness_3$	$Fitness_3$		DIA	$Fitness_2$	$Fitness_3$	$Fitness_3$
	FEZ	$Fitness_3$	$Fitness_3$	$Fitness_2$		FEZ	$Fitness_3$	$Fitness_2$	$Fitness_3$
	VGK	$Fitness_3$	$Fitness_1$	$Fitness_3$		VGK	$Fitness_3$	$Fitness_3$	$Fitness_1$
	EWG	$Fitness_3$	$Fitness_3$	$Fitness_2$		EWG	$Fitness_3$	$Fitness_1$	$Fitness_3$
F3	SPY	$Fitness_3$	$Fitness_1$	$Fitness_3$	F4	SPY	$Fitness_3$	$Fitness_2$	$Fitness_3$
	QQQ	$Fitness_2$	$Fitness_3$	$Fitness_3$		QQQ	$Fitness_3$	$Fitness_3$	$Fitness_3$
	DIA	$Fitness_2$	$Fitness_2$	$Fitness_3$		DIA	$Fitness_1$	$Fitness_3$	$Fitness_2$
	FEZ	$Fitness_3$	$Fitness_3$	$Fitness_1$		FEZ	$Fitness_3$	$Fitness_1$	$Fitness_3$
	VGK	$Fitness_1$	$Fitness_1$	$Fitness_3$		VGK	$Fitness_1$	$Fitness_2$	$Fitness_1$

	EWG	*Fitness*$_3$	*Fitness*$_2$	*Fitness*$_3$		EWG	*Fitness*$_2$	*Fitness*$_3$	*Fitness*$_3$

C. Statistical and trading performance measures.

The statistical and trading performance measures are calculated as shown in table C.1.

Table C.1: Statistical and Trading Performance Measures

STATISTICAL PERFOMANCE	**DESCRIPTION**
Root Mean Squared Error	$RMSE = \sqrt{\frac{1}{N'}\sum_{\tau=t+1}^{t+N'}(E(R_\tau)-Y_\tau)^2}$ *,with* Y_τ *being the actual value and* $E(R_\tau)$ *the forecasted value and* N' *the number of forecasts*
TRADING PERFOMANCE	**DESCRIPTION**
Annualized Return	$R^A = 252 * \frac{1}{N'} * (\sum_{\tau=1}^{N'} R_\tau) - TC^A$ where R_τ is the daily return and TC^A is the annualized transaction cost
Information Ratio	$IR = \frac{R^A}{\sigma^A}$ where σ^A is the annualized volatility